D26 83/1

10/85

TIPS + Tricks von CPU

92 Seiten tolle Tips + Tricks

16 Bit – ein neuer Standard?

6,– DM
6,– sFr
50,– öS
18,– FMK

3. Jahrgang

Spezial-Doppelausgabe Nov./Dez.

Serie

Commodore 64 und VC 20 als universeller Steuerungscomputer

Apple II –

Jetzt arbeitet der „Apple II“ mit 40 Tracks!

Datenübertragung zwischen „Apple II“ und „Commodore“!

Commodore 64

- Komfortables Terminal-Programm für alle DFÜ-Freunde
- Befehlserweiterung für den Profi

Tips + Tricks für MSX!

VC-64 * CPC-464 * VC-20 * Atari * Apple * TI-99 * ZX-Spectrum * MSX *

„Entschnüren“ Sie doch unsere Programmpakete!

„Herr Müller, eigentlich wollten Sie mir ja die tollen Anwender-Programm-Pakete zeigen!“

- Super-Kombi-Discs
- Super-Games
- Utility Special
- Best of . . . Discs
- Appel-Utility
- Atari-Utility

„TIPS & TRICKS von CPU“ hält für Sie ein einmalig günstiges Angebot bereit: Die Redaktion hat sechs tolle Programm-Pakete ausgesucht und versucht Sie nun „loszuwerden“! Über die Preise reden wir jetzt einmal nicht. Diese sind so günstig, daß Sie sie uns doch nicht abnehmen würden. A propos „abnehmen“: Abnehmen können Sie uns, daß wir sechs verschiedene Disc-Programm-Pakete zusammengestellt haben, die von allererster Güte sind. Aber nun ist es doch an der Zeit, den „Vorhang zu lüften“ und die Programme – nebst Preisen – bekannt zu geben:

Bei diesem „KOMBI-DISC-ANGEBOT“ für den **SCHNEIDER CPC 464** handelt es sich um Spiele und Anwenderprogramme – alles nur vom feinsten! Autorennen / Computerschrift / Symbol Swap / Universal-Datei / Keyboard Toolkit / Farbdekodierung / Aladin / Painter / Box-Befehl und BASIC-Maschinen-Kit.
Dieses Paket kostet jetzt nur noch 56 Mark. Kaum zu glauben, was?

Bestell-Nr. US P1

Ja, das sind SUPER-GAMES, die ihrem Namen alle Ehre machen! Folgende MSX-Disketten sind enthalten: Zirkus / Pipeline / Alien Attack. Die Diskette gibt's für ganze 29,50 Mark! „Super Games“, die sich sehen lassen können, gibt es auch für den **ATARI:** Mauerflucht / Lunar-Simulation / Cherry Harry. Preis: „müde“ 24,50 Mark.

Bestell.-Nr. UM P1

Beim „UTILITY SPECIAL“ kommen die **Commodore 64er-Freunde** voll auf ihre Kosten, die sich auf nur 35 „Steine“ belaufen. Was ist drin?
S-Tool 64 / Spritehilfe / Data-Generator / Diskmonitor. Wir meinen, ein tolles Angebot!

Bestell-Nr. UC P1

Die „BEST OF . . . DISCS“ sind ebenfalls für unsere **64er-Freunde** gedacht. Für 35 Mark erhalten Sie folgende „TIPS & TRICKS-BESTSELLER“: Asmon / Re-Assembler / Maskengenerator / Multi BASIC V 3

Bestell-Nr. UC P2

„APPLE-UTILITY“, diese Bezeichnung sagt eigentlich schon alles. Was Sie aber zu einem wahren Super-Preis von 38,50 Mark geboten bekommen, wird Sie erstaunen! Dateiverwaltung / Disk / Cheque / DOS Change / Digital Sound Editor.

Bestell-Nr. UA P1

Die **„ATARI UTILITY“** *ist* ebenfalls Spitze, wir können dieses Disketten-Programm mit gutem Gewissen empfehlen! Und das zu einem Preis, den fast schon die Leer-Diskette kostet: 25 Mark! Zeichensteller / Zeicheneditor / Monitorprogramm.

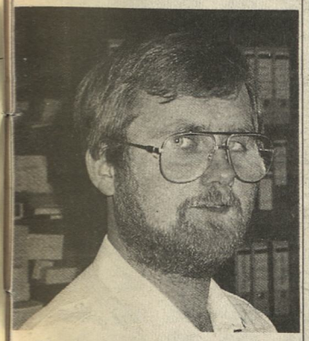
Bestell-Nr. UJ P1

Da heißt es, nicht lange fackeln!

ZUGREIFEN!

Liebe Leser der „Tips u. Tricks von CPU“

Aufgrund der zahlreichen Zuschriften, die die CPU-Redaktion erreichten, gehen wir davon aus, daß die neue „TIPS & TRICKS von CPU“ bei Ihnen gut angekommen ist. Dies freut uns sehr! Wir wollen uns jedoch keineswegs auf den „geernteten Lorbeeren“ ausruhen; vielmehr werden künftig noch mehr Tips & Tricks, Programm- und Textbeiträge und dergleichen publiziert. Sie sehen also, die Qualität der Berichte und Programme geht uns vor der Quantität. In diesem Punkt sind wir uns doch einig, oder? A propos „wir“: Mit „Wir“ meinen wir den „verstärkten Einsatz“ des Lesers bei der Gestaltung der CPU. Deshalb finden Sie in diesem Heft auch einige Wettbewerbe, bei denen der Leser selbst „das Programm bestimmen“ soll – und zu gewinnen gibt's auch 'was, versteht sich! Die Redaktion möchte auf diesem Wege versuchen, ein noch besseres Feedback, sprich: Kontakt Leser – CPU – erreichen. Wir hoffen, Sie helfen uns dabei!



Ihr
Siegfried Görk
Chefredakteur CPU

PS: Das Team – „Tips & Tricks von CPU“ – dankt für die zahlreichen Einsendungen! Macht weiter so!

* * * * * * *

„Tips u. Tricks von CPU“ – Der Inhalt:

UTILITY — Hilfsprogramme für den Programmierer

		Seite
Apple II:	„Big Spender“	4
CPC-464:	Backup-Programm	10
CPC-464:	Basic-Befehlserweiterung/Ellipse	13
CPC-464/664:	Directory-Ausgabe auf dem Drucker	17
MSX:	Zeichensatzkonverter für Epson-Drucker	21
MSX:	„Screensave“	23
Atari:	Zeichensteller für GR. 12 + 13	24
TI-99:	Stringkit (Basic-Erweiterung)	28
ZX-Spectrum:	MD-Copierer	36
VC-20:	Directory-Programm	39
C-64:	Hardcopy-Routine für den C-64 + MPS 801	41
C-64:	Utility-Routine: Autonumber	42

USER — Programme für den Anwender

ZX-Spectrum:	Disassembler	47
C-64:	Terminalprogramm	53
Apple II/C-64:	Datenübertragung zwischen zwei Rechnern	61
Atari:	Error-Helper (Teil 2)	69

„Tips und Tricks“-Extra

MSX:		70/71
SVI:		73
CPC-464:		74/75
VC-20		76/77
Apple II:		77
Allgemein:	Sortierverfahren	80
Allgemein:	„Logoleien mit dem Computer“	81

HARDWARE

VC-20/C-64:	Universelles Bussystem/Bauleitung (Teil 2)	84

Rubriken

Computer-Club-Vorstellung:		9/12
Impressum:		27
Im Test:	Matrix-Drucker: „Star SG-10“	44
	Neue 16-Bit-Rechner im Vergleich	45
Vorschau:	In der nächsten „Tips u. Tricks von CPU“	68/87
Infos:	Programmiersprachen im Vergleich	79
Software-Service:	Gesamtkatalog	88/89
CPU-Basar:	Kleinanzeigen	90
Autorenwettbewerb:	600 DM zu gewinnen!	91

Apple Programm: „Big Spender“

Mit Hilfe dieses Programmes lassen sich beim Apple-Laufwerk, 40, anstatt der normalen 35 Tracks initialisieren. Im Gegensatz zu vielen anderen Programmen ist es außerdem möglich schon beschriebene Disketten um 5 Tracks zu erweitern, ohne daß dabei Daten verlorengehen.
Dazu kann Big Spender auch noch 11 Sektoren zur freien Verfügung stellen, die normaler Weise vom DOS belegt sind. Ergebnis: Ein Gewinn von 23 KBYTE (17,8 %).

Zu beachten ist:

Original Apple-Laufwerke können leider keine 40 Tracks bearbeiten (maximal 36), aber viele der modernen Nachbauten sind dazu in der Lage. Vergewissern Sie sich im Handbuch, ob dies bei Ihrem Laufwerk möglich ist, bevor Sie das Programm eingeben.
Big Spender arbeitet nur mit DOS 3.3, im Gegensatz zu den erstellten Disketten, die auch mit den Versionen Q-DOS und DOS 3.4 arbeiten. Vor dem ersten Start sollten Sie von Ihrer Diskette eine Sicherheitscopy anfertigen, da durch Tippfehler die Diskette zerstört werden kann.
Das Kopieren einer 40 Track-Diskette ist natürlich nicht mit Standard-Kopierprogrammen wie COPYA (von Apple) oder COPYB (von F. Brall) möglich, hierfür sind spezielle Programme wie LOCKSMITH 5.0 nötig. BIG SPENDER ist so ausgelegt, daß Sie jederzeit mit RESET oder dem Befehl & in das Programm zurück gelangen können. Aus Sicherheitsgründen können die Tracks 36-40 nur einmal formatiert werden.
Das Programm ist als HEXLISTING abgedruckt und wird mit dem Apple Monitor (Call -151) eingegeben. Nach der Eingabe wird es mit BSAVE BIG SPENDER ,A$803,L$D66 abgespeichert.

23 K-Byte-Diskettenerweiterung auf 40 Spuren (!)

```
0800- 00 00 00 A9 4C 8D F5 03
0808- A9 03 8D F6 03 8D F2 03
0810- A9 08 8D F7 03 8D F3 03
0818- 49 A5 8D F4 03 20 2F FB
0820- A9 01 85 F1 A9 04 8D 7D
0828- 0D 20 3D 0D A9 A7 85 06
0830- A9 10 85 07 A9 49 85 08
0838- A9 11 85 09 A0 0B 20 90
0840- 0D 20 E3 03 84 06 85 07
0848- A0 08 B1 06 8D C3 10 A9
0850- 00 91 06 85 08 C8 B1 06
0858- 8D C4 10 A9 20 91 06 85
0860- 09 A0 0F 18 B1 06 4A 4A
0868- 4A 4A 8D C7 10 09 B0 8D
0870- 19 11 8D 2D 11 C8 B1 06
0878- 8D C6 10 09 B0 8D 16 11
0880- 8D 2A 11 A9 00 8D 5E AA
0888- A2 0D 20 A6 0D A2 00 BD
0890- 33 11 F0 06 20 F0 FD E8
0898- D0 F5 20 19 10 20 3D 0E
08A0- A9 01 85 22 20 BA 0D 20
08A8- F3 0D A2 00 86 24 BD C4
08B0- 13 F0 0C C9 C0 90 02 29
08B8- FF 20 F0 FD E8 D0 EF A9
08C0- BF 8D F2 03 A9 9D 8D F3
08C8- 03 A9 38 8D F4 03 4C D3
08D0- 03 A9 0D 85 25 A9 19 85
08D8- 24 20 62 0E A9 01 8D BE
08E0- 10 20 70 0E 90 10 20 A2
08E8- 0E A2 00 A9 15 20 A6 0D
08F0- 20 42 FC A9 07 60 A9 00
08F8- 8D B9 10 8D BA 10 A5 25
0900- 48 A5 24 48 A0 34 B1 08
0908- 0A 0A 18 69 36 8D C5 10
0910- A0 38 8C B7 10 A2 08 8E
0918- BF 10 B1 08 8D C2 10 0E
0920- C2 10 90 23 AD B9 10 69
0928- 00 8D B9 10 90 03 EE BA
0930- 10 68 85 24 68 85 25 48
0938- 20 C1 FB A5 24 48 AD BA
0940- 10 AE B9 10 20 24 ED CE
0948- BF 10 D0 D3 EE B7 10 AC
0950- B7 10 CC C5 10 D0 BE 68
0958- 68 A9 07 60 20 F4 0E 20
0960- 62 0E A9 01 8D BE 10 20
0968- 70 0E 90 03 4C E6 08 18
0970- A0 09 B1 08 A0 40 71 08
0978- C8 71 08 D0 2B A9 E0 91
0980- 08 88 A9 FF 91 08 A0 09
0988- A9 01 91 08 A0 06 B1 08
0990- 8D BD 10 A9 02 8D BE 10
0998- 20 70 0E B0 CF A9 0D 85
09A0- 25 A9 19 85 24 4C F6 08
09A8- A2 00 A9 16 20 A6 0D BD
09B0- 47 12 F0 0C C9 C0 90 02
09B8- 29 FF 20 F0 FD E8 D0 EF
09C0- 20 AE 0D 4C E9 08 20 F4
09C8- 0E 20 62 0E A9 01 8D BE
09D0- 10 20 70 0E 90 06 20 E6
09D8- 08 4C A4 08 20 3D 0D A2
09E0- 00 A9 04 20 A6 0D BD 2E
09E8- 13 F0 0C C9 C0 90 02 29
09F0- FF 20 F0 FD E8 D0 EF A2
09F8- 1A A9 04 20 A6 0D A0 34
0A00- B1 08 8D BB 10 8D C0 10
0A08- AA A9 00 20 24 ED A9 00
0A10- 85 50 85 51 20 47 0F A9
0A18- 06 A2 25 20 A6 0D 20 12
0A20- 0F C9 53 D0 03 4C A4 08
0A28- A5 51 D0 E2 A5 50 F0 DE
0A30- C9 29 B0 DA CD BB 10 90
0A38- D5 F0 D3 A9 13 A2 00 20
0A40- A6 0D BD EC 12 F0 0C C9
0A48- C0 90 02 29 FF 20 F0 FD
0A50- E8 D0 EF A2 00 BD CB 12
0A58- F0 0C C9 C0 90 02 29 FF
0A60- 20 F0 FD E8 D0 EF 20 47
0A68- 0F C9 A0 D0 F9 A9 12 A2
0A70- 00 20 A6 0D 20 42 FC A5
0A78- 50 18 ED BB 10 8D B8 10
0A80- CE BB 10 A0 06 B1 08 8D
0A88- BD 10 85 41 A0 03 91 06
0A90- A0 55 A9 00 99 00 BC 88
0A98- 10 FA 99 00 BB 88 D0 FA
0AA0- A0 01 B1 06 AA BD 88 C0
0AA8- 20 47 0F EE BB 10 CE B8
0AB0- 10 10 03 4C 42 0B A2 1A
0AB8- A9 04 20 A6 0D A9 00 AE
0AC0- BB 10 20 24 ED AD BB 10
0AC8- A0 04 91 06 A9 00 A0 0C
0AD0- 91 06 20 E3 03 20 D9 03
0AD8- A0 01 B1 06 AA AD C6 10
0AE0- C9 01 F0 05 BD 8B C0 B0
0AE8- 03 BD 8A C0 BD 89 C0 AD
0AF0- BB 10 85 44 A9 AA 85 3E
0AF8- A9 28 85 45 20 0D BF B0
0B00- 16 A9 30 8D C1 10 38 CE
0B08- C1 10 F0 0B 20 44 B9 B0
0B10- F5 A5 2D D0 F1 90 89 BD
0B18- 88 C0 A2 00 A9 15 20 A6
0B20- 0D A2 00 BD E6 10 F0 0C
0B28- C9 C0 90 02 29 FF 20 F0
0B30- FD E8 D0 EF AD BB 10 CD
0B38- C0 10 D0 06 20 AE 0D 4C
0B40- A4 08 A2 1A A9 04 20 A6
0B48- 0D A9 00 AE BB 10 20 24
0B50- ED A2 05 A9 FF 20 A8 FC
0B58- CA 10 F8 A2 1A A9 04 20
0B60- A6 0D A9 00 EE BB 10 AE
0B68- BB 10 8E C5 10 20 24 ED
0B70- 20 62 0E A9 01 8D BE 10
0B78- 20 70 0E 90 06 20 E6 08
0B80- 4C A4 08 A0 34 AD C5 10
0B88- 91 08 0A 0A 18 69 34 A8
0B90- A9 FF 91 08 C8 91 08 CE
0B98- C5 10 AD C5 10 CD C0 10
0BA0- D0 E8 A9 02 8D BE 10 20
0BA8- 70 0E 90 06 20 E6 08 4C
0BB0- A4 08 A9 0B A2 00 20 A6
0BB8- 0D BD A1 13 F0 0C C9 C0
0BC0- 90 02 29 FF 20 F0 FD E8
0BC8- D0 EF A9 0B 85 25 A9 10
0BD0- 85 24 20 F6 08 20 AE 0D
0BD8- 4C A4 08 20 3D 0D A9 02
0BE0- A2 00 20 A6 0D BD 8A 12
0BE8- F0 0C C9 C0 90 02 29 FF
0BF0- 20 F0 FD E8 D0 EF 20 47
0BF8- 0F C9 A0 D0 F9 20 58 FC
0C00- A9 0A A2 0A 20 A6 0D A2
0C08- 00 BD 88 13 F0 0C C9 C0
0C10- 90 02 29 FF 20 F0 FD E8
0C18- D0 EF A2 00 BD 54 13 C9
0C20- C0 90 02 29 FF 20 F0 FD
0C28- E8 E0 25 D0 EF A9 00 85
0C30- 50 85 51 2C 10 C0 20 12
0C38- 0F C9 53 D0 03 4C A4 08
0C40- A5 51 D0 BC A5 50 F0 B8
0C48- C9 23 90 B4 C9 29 B0 B0
0C50- 8D BB 10 A2 00 A9 12 20
0C58- A6 0D BD EC 12 F0 0C C9
0C60- C0 90 02 29 FF 20 F0 FD
0C68- E8 D0 EF A2 00 BD CB 12
0C70- F0 0C C9 C0 90 02 29 FF
0C78- 20 F0 FD E8 D0 EF 20 47
0C80- 0F C9 A0 D0 F9 A2 00 A9
0C88- 11 20 A6 0D 20 42 FC AD
0C90- B5 AE 8D B9 10 AD FE BE
0C98- 8D BA 10 A9 00 8D C4 B3
0CA0- AD BB 10 8D EF B3 8D FE
0CA8- BE C9 24 90 49 E9 23 0A
0CB0- 0A 18 69 8C 8D B5 AE A9
0CB8- EF 8D BC AD A9 B3 8D BD
0CC0- AD A2 05 BD FD 10 9D B0
0CC8- B3 CA 10 F7 20 FF 0C A9
0CD0- 00 85 48 AD B9 10 8D B5
0CD8- AE AD BA 10 8D FE BE A2
0CE0- 05 BD 03 11 9D B0 B3 CA
0CE8- 10 F7 A9 F6 8D BC AD A9
0CF0- B7 8D BD AD D0 03 20 FF
0CF8- 0C 2C 10 C0 4C 64 0F A9
0D00- 12 85 67 A9 14 85 68 A9
0D08- 15 85 B0 A9 69 85 AF AD
0D10- C3 10 A0 08 91 06 C8 AD
0D18- C4 10 91 06 A9 09 A0 11
0D20- 20 3A DB A9 00 A0 08 91
0D28- 06 C8 A9 20 91 06 A9 01
```

```
0D30- 85 67 A9 08 85 68 85 B0
0D38- A9 03 85 AF 60 A9 28 8D
0D40- B6 10 A2 00 BD 77 10 85
0D48- 1A E8 BD 77 10 85 19 E8
0D50- A0 00 8C B4 10 C8 8C B5
0D58- 10 AC B5 10 B1 19 AC B4
0D60- 10 91 19 EE B4 10 EE B5
0D68- 10 AD B5 10 C9 28 D0 E9
0D70- A9 A0 AC B4 10 91 19 18
0D78- A5 19 65 1A C9 04 D0 C4
0D80- CE B6 10 D0 BD A2 05 BD
0D88- A7 10 09 80 8D 7D 0D 60
0D90- A2 07 B1 06 0A 6E B3 10
0D98- CA 10 F9 AD B3 10 09 80
0DA0- 91 08 88 10 EB 60 86 24
0DA8- 85 25 20 C1 FB 60 2C 10
0DB0- C0 AD 00 C0 10 FB 2C 10
0DB8- C0 60 20 58 FC A9 05 A2
0DC0- 08 20 A6 0D A2 00 BD 57
0DC8- 11 F0 0C C9 C0 90 02 29
0DD0- FF 20 F0 FD E8 D0 EF A9
0DD8- 05 A2 10 20 A6 0D A9 00
0DE0- AE C7 10 20 24 ED A2 19
0DE8- 86 24 A9 00 AE C6 10 20
0DF0- 24 ED 60 A9 13 A2 16 20
0DF8- A6 0D 20 9C FC 20 0C FD
0E00- 20 F0 FD C9 B0 D0 03 20
0E08- AF 0F C9 B1 D0 03 20 5C
0E10- 09 C9 B2 D0 03 20 C6 09
0E18- C9 B3 D0 03 20 DB 0B C9
0E20- B4 D0 03 20 D1 08 C9 B5
0E28- D0 03 20 64 0F C9 B6 D0
0E30- C2 A9 00 85 22 A9 04 8D
0E38- 7D 0D 4C 3D 0D A9 4C 20
0E40- B2 A5 D0 01 60 A2 00 A9
0E48- 0A 20 A6 0D BD 6A 12 F0
0E50- 0C C9 C0 90 02 29 FF 20
0E58- F0 FD E8 D0 EF 68 68 4C
0E60- D3 03 A9 11 8D BB 10 A9
0E68- 00 8D BC 10 8D BD 10 60
0E70- AD BD 10 A0 03 91 06 AD
0E78- BB 10 A0 04 91 06 AD BC
0E80- 10 A0 05 91 06 A5 08 A0
0E88- 08 91 06 C8 A5 09 91 06
0E90- AD BE 10 A0 0C 91 06 20
0E98- E3 03 20 D9 03 A9 00 85
0EA0- 48 60 A2 0D A9 16 20 A6
0EA8- 0D A0 0D B1 06 C9 10 F0
0EB0- 2A C9 40 D0 13 A2 00 BD
0EB8- C8 10 F0 32 C9 C0 90 02
0EC0- 29 FF 20 F0 FD E8 D0 EF
0EC8- A2 04 BD C8 10 F0 1F C9
0ED0- C0 90 02 29 FF 20 F0 FD
0ED8- E8 D0 EF A2 00 BD D4 10
0EE0- F0 0C C9 C0 90 02 29 FF
0EE8- 20 F0 FD E8 D0 EF 20 AE
0EF0- 0D A9 07 60 A2 00 A9 16
0EF8- 20 A6 0D BD 22 12 F0 0C
0F00- C9 C0 90 02 29 FF 20 F0
0F08- FD E8 D0 EF 20 AE 0D 4C
0F10- E9 08 20 2C D5 AD 00 02
0F18- C9 53 F0 2A 86 B8 84 B9
0F20- 46 D8 A2 00 BD 00 02 F0
0F28- 0B C9 30 90 19 C9 3A B0
0F30- 15 E8 D0 F0 20 B1 00 F0
0F38- 0D 20 59 D5 E6 B8 E6 B9
0F40- 20 67 DD 20 52 E7 60 AD
0F48- 00 C0 C9 A0 F0 0D 29 DF
0F50- C9 D3 F0 08 C9 9B F0 04
0F58- 2C 10 C0 60 68 68 68 68
0F60- 68 4C A4 08 20 58 FC A9
0F68- 74 8D 5E 9D A9 0F 8D 5F
0F70- 9D 4C BF 9D A9 A7 8D 5E
0F78- 9D A9 1F A0 11 20 3A DB
0F80- A2 00 BD 16 13 F0 0C C9
0F88- C0 90 02 29 FF 20 F0 FD
0F90- E8 D0 EF 20 0C FD A9 3C
0F98- 8D 5E 9D A9 D4 8D 5F 9D
0FA0- A9 01 85 22 4C A4 08 20
0FA8- AE 0D D0 EA 4C A7 0F A2
0FB0- 10 A9 05 20 A6 0D 20 0C
0FB8- FD C9 8D F0 24 C9 B1 90
0FC0- EE C9 B7 F0 02 B0 E8 8D
0FC8- 19 11 8D 2D 11 49 B0 8D
0FD0- 6A AA 8D C7 10 0A 0A 0A
0FD8- 0A A0 01 91 06 A0 0F 91
0FE0- 06 AE C7 10 A9 00 20 24
0FE8- ED A9 19 85 24 20 0C FD
0FF0- C9 8D F0 1A C9 B1 F0 04
0FF8- C9 B2 D0 ED 8D 16 11 8D
1000- 2A 11 49 B0 8D 68 AA A0
1008- 02 91 06 8D C6 10 AE C6
1010- 10 A9 00 20 24 ED A9 07
1018- 60 A2 00 BD ED 13 F0 06
1020- 20 F0 FD E8 D0 F5 20 0C
1028- FD 29 DF C9 CA F0 44 C9
1030- CE F0 04 C9 8D D0 EF A9
1038- DF 8D B8 08 8D B9 09 8D
```



```
1040- F0 09 8D 4C 0A 8D 5F 0A
1048- 8D 2D 0B 8D C3 0B 8D EF
1050- 0B 8D 13 0C 8D 24 0C 8D
1058- 64 0C 8D 77 0C 8D D0 0D
1060- 8D 56 0E 8D C1 0E 8D D4
1068- 0E 8D E7 0E 8D 05 0F 8D
1070- 8C 0F 60 A9 FF D0 C2 07
1078- D0 07 50 06 D0 06 50 05
1080- D0 05 50 04 D0 04 50 07
1088- A8 07 28 06 A8 06 28 05
1090- A8 05 28 04 A8 04 28 07
1098- 80 07 00 06 80 06 00 05
10A0- 80 05 00 04 80 04 00 52
10A8- 74 04 22 A2 04 12 82 82
10B0- CA 74 B0 00 00 00 00 00
10B8- 00 00 00 00 00 00 00 00
10C0- 00 00 00 00 00 00 00 00
10C8- C9 AF CF A0 C5 D2 D2 CF
10D0- D2 AE 87 00 D7 D2 C9 D4
10D8- C5 A0 D0 D2 CF D4 C5 C3
10E0- D4 C5 C4 AE 87 00 C5 D2
10E8- D2 CF D2 A0 C9 CE A0 C6
10F0- CF D2 CD C1 D4 C5 D2 C9
10F8- CE C7 AE 87 00 D3 CB C3
1100- C1 D2 D4 C5 CD D5 CC CF
1108- D6 8D 84 C9 CE C9 D4 C8
1110- C5 CC CC CF AC C4 00 AC
1118- D3 00 AC D6 B0 8D 00 8D
1120- 84 C3 C1 D4 C1 CC CF C7
1128- AC C4 00 AC D3 00 AC D6
1130- B0 8D 00 20 02 09 07 20
1138- 13 10 05 0E 04 05 12 20
1140- 8D 8D 8D C4 D5 D2 C3 C8
1148- A0 CA AE A0 C4 C5 A0 C8
1150- C1 C1 D3 AE 8D 8D 00 B0
1158- A9 A0 D3 EC EF F4 A0 AE
1160- AC A0 C4 F2 E9 F6 E5 A0
1168- AE 8D 8D A0 A0 A0 A0 A0
1170- A0 A0 A0 B1 A9 A0 B1 B1
1178- A0 E5 F8 F4 F2 E1 A0 D3
1180- E5 E3 F4 EF F2 E5 EE 8D
1188- 8D A0 A0 A0 A0 A0 A0 A0
1190- A0 B2 A9 A0 E5 F8 F4 F2
1198- E1 A0 D4 F2 E1 E3 EB F3
11A0- 8D 8D A0 A0 A0 A0 A0 A0
11A8- A0 A0 B3 A9 A0 E9 EE E9
11B0- F4 E9 E1 EC E9 F3 E9 E5
11B8- F2 A0 C4 E9 F3 EB 8D 8D
11C0- A0 A0 A0 A0 A0 A0 A0 A0
11C8- B4 A9 A0 D3 F0 E5 E9 E3
11D0- E8 E5 F2 F0 EC E1 F4 FA
11D8- A0 DF DF DF A0 D3 E5 E3
11E0- F4 8D 8D A0 A0 A0 A0 A0
11E8- A0 A0 A0 B5 A9 A0 E3 E1
11F0- F4 E1 EC EF E7 8D 8D A0
11F8- A0 A0 A0 A0 A0 A0 A0 B6
1200- A9 A0 E2 E5 E5 EE E4 E5
1208- EE 8D 8D A0 A0 A0 A0 A0
1210- A0 A0 A0 A0 A0 A0 A0 C9
1218- E8 F2 E5 A0 D7 E1 E8 EC
1220- BA 00 D3 F4 E5 E3 EB A0
1228- E5 E9 EE E5 A0 C4 E9 F3
1230- EB E5 F4 F4 E5 A0 E9 EE
1238- A0 E4 E1 F3 A0 CC E1 F5
1240- E6 F7 E5 F2 EB AE 00 C4
1248- E9 E5 F3 E5 A0 D3 E5 E3
1250- F4 EF F2 E5 EE A0 F3 E9
1258- EE E4 A0 E2 E5 F2 E5 E9
1260- E4 F3 A0 E6 F2 E5 E9 AE
1268- 87 00 C1 F0 F0 EC E5 AD
1270- F3 EF E6 F4 A0 ED F5 F3
1278- F3 A0 E1 EE F7 E5 F3 E5
1280- EE E4 A0 F3 E5 E9 EE AE
1288- 87 00 57 41 52 4E 55 4E
1290- 47 61 87 A0 C4 E9 E5 F3
1298- E5 A0 D2 EF F5 F4 E9 EE
12A0- E5 A0 F6 E5 F2 EE E9 E3
12A8- E8 F4 E5 F4 8D 8D E1 EC
12B0- EC E5 A0 C4 E1 F4 E1 A0
12B8- E1 F5 E6 A0 E4 E5 F2 A0
12C0- C4 E9 F3 EB E5 F4 F4 E5
12C8- AE 8D 8D D3 BD F3 F4 EF
12D0- F0 F0 E5 EE A0 A0 A0 A0
12D8- F3 F0 E1 F4 E9 E5 BD F7
12E0- E5 E9 F4 E5 F2 E7 E5 E8
12E8- E5 EE AE 00 C9 F3 F4 A0
12F0- E4 E9 E5 A0 F2 E9 E3 E8
12F8- F4 E9 E7 E5 A0 C4 E9 F3
1300- EB E5 F4 F4 E5 A0 E9 ED
1308- A0 CC E1 F5 E6 F7 E5 F2
1310- EB BF 87 8D 8D 00 C4 F2
1318- F5 E3 EB E5 EE A0 D3 E9
1320- E5 A0 E5 E9 EE E5 A0 D4
1328- E1 F3 F4 E5 BA 00 C4 E9
1330- E5 F3 E5 F2 A0 C4 E9 F3
1338- EB E5 F4 F4 E5 A0 E8 E1
1340- F4 A0 EA E5 F4 FA F4 A0
1348- DF DF A0 D4 F2 E1 E3 EB
```

```
1350- F3 AE 8D 8D D7 E9 E5 F6
1358- E9 E5 EC A0 D4 F2 E1 E3
1360- EB F3 A0 F7 F5 EE F3 E3
1368- E8 E5 EE A0 D3 E9 E5 A0
1370- A8 ED E1 F8 A0 B4 B0 A9
1378- BA 8D 8D D3 BD F3 F4 EF
1380- F0 F0 E5 EE 8D 8D 8D 00
1388- C9 EE E9 F4 E9 E1 EC E9
1390- F3 E9 E5 F2 A0 C4 E9 F3
1398- EB E5 F4 F4 E5 AE 8D 8D
13A0- 00 D3 E9 E5 A0 E8 E1 E2
13A8- E5 EE A0 EA E5 F4 FA F4
13B0- A0 DF DF DF A0 F3 E5 E3
13B8- F4 EF F2 E5 EE A0 E6 F2
13C0- E5 E9 AE 00 CD E9 F4 A0
13C8- A2 A6 A2 A0 E7 E5 E8 E5
13D0- EE A0 D3 E9 E5 A0 FA F5
13D8- F2 F5 E3 EB A0 E9 EE F3
13E0- A0 D0 F2 EF E7 F2 E1 ED
13E8- ED AE 8D 8D 00 8D 8D C8
13F0- C1 D4 A0 C9 C8 D2 A0 C1
13F8- D0 D0 CC C5 A0 CB CC C5
1400- C9 CE D3 C3 C8 D2 C9 C6
1408- D4 A0 DB CA AF CE DD A0
1410- BA 00 07 08 0A 00 97 00
1418- 17 08 14 00 BA 22 04 43
1420- 41 54 41 4C 4F 47 22 00
1428- 42 08 FF FF B2 08 08 08
1430- 08 08 08 08 08 08 08 08
1438- 08 08 08 08 08 08 20 20
1440- 20 20 20 20 20 20 20 20
1448- 20 20 20 20 20 20 20 0A
1450- 0A 0A 00 69 08 FF FF B2
1458- 08 08 08 08 08 08 08 08
1460- 08 08 08 08 08 08 08 08
1468- 08 2A 2A 2A 2A 2A 2A 2A
1470- 2A 2A 2A 2A 2A 2A 2A 2A
1478- 20 00 8F 08 FF FF B2 08
1480- 08 08 08 08 08 08 08 08
1488- 08 08 08 08 08 08 08 08
1490- 2A 20 20 20 20 20 20 20
1498- 20 20 20 20 20 20 2A 00
14A0- B5 08 FF FF B2 08 08 08
14A8- 08 08 08 08 08 08 08 08
14B0- 08 08 08 08 08 08 2A 20
14B8- 42 49 47 20 53 50 45 4E
14C0- 44 45 52 20 2A 00 DC 08
14C8- FF FF B2 08 08 08 08 08
14D0- 08 08 08 08 08 08 08 08
14D8- 08 08 08 08 2A 20 20 20
14E0- 20 20 20 20 20 20 20 20
14E8- 20 20 2A 20 00 03 09 FF
14F0- FF B2 08 08 08 08 08 08
14F8- 08 08 08 08 08 08 08 08
1500- 08 08 08 2A 20 4A 2E 20
1508- 44 45 20 48 41 41 53 2E
1510- 20 2A 20 00 2A 09 FF FF
1518- B2 08 08 08 08 08 08 08
1520- 08 08 08 08 08 08 08 08
1528- 08 08 2A 20 20 20 20 20
1530- 20 20 20 20 20 20 20 20
1538- 2A 20 00 53 09 FF FF B2
1540- 08 08 08 08 08 08 08 08
1548- 08 08 08 08 08 08 08 08
1550- 08 2A 2A 2A 2A 2A 2A 2A
1558- 2A 2A 2A 2A 2A 2A 2A 2A
1560- 0A 0A 0A 00 00 00 00 00
1568- 00 00
```

Die Gewinner von CPU-Happy aus Heft Nr. 8

1. Preis:
10 Kassetten aus unserem Software-Service:
Udo Maaß, Bad Herrenalb 3

2. Preis:
3 Kassetten aus unserem Software-Service:
Peter Mayrhofer, Wien

3. Preis:
2 Kassetten aus unserem Software-Service:
Jürgen Krätzschmar, Hemmingen

4.-18. Preis
1 Kassette aus unserem Software-Service:
Herbert Wolfsteller, Kiel
Rudolf Mensing, Kaltenkirchen
Harald Schüßler, Kleve
Lutz Keim, Berlin 41
Kai Meyer, Bebensee
Richard Meyer, Bebensee
Gerhard Kloss, Lübeck
Hans-Martin Will, Waiblingen
Peter Diebold, Puchheim
J.G.P. Sijm, Zwanenburg (Holland)
Flemming Kaufmann, Sønderborg (Dänemark)
H.-J. Schaper, Springe 6
Josef Kramer, Holzheim/Riedheim
Wolfgang Heitz, Burgheim
Robert-Jan Milleker, Bonn

* * * *

An alle Leser von „Tips u. Tricks v. CPU“! Auch in Zukunft gibt es bei uns viel zu gewinnen. – Mehr darüber auf den nächsten Seiten.

Spectrum-User-Club-Wuppertal

Der Spectrum-User-Club-Wuppertal – mit fast 300 Mitgliedern wohl der größte, überregional arbeitende Zusammenschluß von ZX Spectrum-Benutzern – feiert Geburtstag. Im August 1985 besteht unser Club nun genau 2 Jahre. In dieser Zeit ist es uns gelungen, einige hundert Seiten voller Informationen über Hard- und Software, Literatur, Tips & Tricks usw. im Rahmen unserer monatlich erscheinenden Club-Zeitschrift an den Mann (Frau) zu bringen. Gerade in der heutigen Zeit, in der der ZX Spectrum nicht unbedingt im Mittelpunkt des Interesses steht (was sich leider auch bei den Computerzeitschriften bemerkbar macht), zeigt uns die Resonanz unserer Mitglieder immer wieder, wie wichtig ein User-Club sein kann. Gerade in mittleren und kleinen Orten dieser Republik ist es nicht leicht, einen Ansprechpartner zu finden, der in der Lage ist, Hilfestellung bei den verschiedensten Problemen zu geben. Auch der interne Software-Service – eine monatlich erscheinende Kassette mit von Club-Mitgliedern erstellter Software – erfreut sich großer Beliebtheit. Wir hoffen, daß die nächsten 2 Jahre ebenso erfolgreich verlaufen wie die vergangenen.

Wer weitere Informationen wünscht, kann diese gegen Rückporto anfordern bei:
Rolf Knorre, Postfach 200102, 5600 Wuppertal 2.
Wir sind für jeden Kontakt dankbar.

Arbeitsgemeinschaft der C-64-User

Die Arbeitsgemeinschaft der C64-User existiert bereits seit Anfang April dieses Jahres. Ziel ist es, den Informationsaustausch zwischen Commodore-64-Anwendern anzukurbeln. Das soll hauptsächlich auf überregionaler Ebene geschehen, da der Club möglichst viele Interessenten ansprechen will. In den letzten Monaten widmete man sich hauptsächlich der Konzeption des Vereinsorgans MATRIX 64. Dieses Magazin soll alle vier Monate über die Aktivitäten der AG 64 informieren. Das Magazin wird für ,-- DM im Postversand verrieben. Bei einer Mitgliedschaft der AG 64 (Jahresbeitrag 0,-- DM) erhält das Clubmitglied zusätzlich zu einem Jahresabo MATRIX 64, regelmäßig Listen der Buch- und Softwarebibliothek zugeschickt.
Für den Herbst plant der Club im Zusammenhang mit dem Erscheinen der ersten Ausgabe

* * * * * * * * * * * *
* *
* *An alle Computer-* *
* *vereinigungen! –* *
* *Nutzen Sie die* *
* *kostenlose* *
* *Clubvorstellung in* *
* *„Tips u. Tricks* *
* *von CPU“* *
* *
* * * * * * * * * * * *

von MATRIX 64 eine große Mitgliederwerbeaktion, in deren Verlauf die bisher recht niedrige Mitgliederzahl von 10 Computerfreaks gesteigert werden soll.

Kontaktadresse:
Arbeitsgemeinschaft der C64 User
c/o Klaus Vill
Albertusstraße 45/1
8901 Biberbach

Club der EDV-Füchse

Bei diesem Club handelt es sich um eine Interessengemeinschaft von Computerbenutzern der verschiedensten Systeme mit der Zielsetzung, durch Sammelbestellungen über den Großhandel zu wesentlich günstigeren Preisen einkaufen zu können.
Gerade im Sektor Datenträger (Disketten und Datencassetten) konnten schon beste Erfolge erzielt werden.
Doch auch bei kompletten Systemen wie zum Beispiel dem Commodore PC10 wurden schon erstaunlich niedrige Preise erhandelt.
Aber nicht nur im Zusammenschluß zwecks Sammelbestellungen liegt das Ziel dieses Clubs.
Ein weiteres Bestreben dieser Vereinigung liegt darin, Benutzer eines gemeinsamen Systems, die relativ nah beieinander wohnen, aufeinander aufmerksam zu machen.
Die Clubführung bittet deshalb jeden der beitreten will, eine genaue Beschreibung seiner Person und seines Systemes sowie seiner bevorzugt benutzten Programme/Sprachen zu senden, so daß diese Daten in die Club-Datenbank eingehen können, die automatisch überprüft, ob ein neues Mitglied in der Nähe eines schon bekannten wohnt.
Ist dies der Fall werden ebenfalls automatisch beide Computer-User aufeinander aufmerksam gemacht.
Ein letzter wesentlicher Punkt des Clubs besteht in der Versendung einer kritischen Info-Zeitschrift, der auch eine aktuelle Mitgliederpreisliste mit gerade aktuellen, besonders guten Angeboten beiliegt.
Für diese Info wird ein Unkostenbeitrag von 1,40 DM (Versand-/Materialkosten) erhoben.
Will man diesen Betrag nicht investieren, kann man auch auf die Infos verzichten; man ist nicht gezwungen, sie zu „abonnieren“.
Dennoch glaubt die Clubführung, daß diese einen guten Zusatz zu einer guten Computerzeitschrift darstellen.
Sollten Sie Interesse an dieser Vereinigung haben, so fordern Sie bitte nähere Informationen schriftlich mit einem frankierten Rückumschlag bei
Club der EDV-Füchse
Am Stiegelberg 15
7552 Durmersheim
an.
Weitere Clubvorhaben:
Eine Mailbox ist geplant, ebenso wie ein Telefonanrufbeantworter, auf dem die neuesten Informationen auch für Nichtakustikkopplerbesitzer zugänglich sind.
(Gedacht als Alternative für Mitglieder, die kein Info bekommen).

Das Programm „BACKUP" dient zur Erstellung von Sicherheitskopien von 3" Disketten auf Kassetten

Backup-Programm für den CPC-464

Vorteile gegenüber konventionellen Sicherungsmaßnahmen:

1. Sicherheitskopien auf 3" Disketten sind teuer und nutzen die Auswurfmechanik des Laufwerks stark ab.
2. Sicherheitskopien von Diskettenfiles auf Kassette sind i.a. umständlich zu erstellen, nehmen viel Speicherplatz ein und sind nur langsam zu laden.

Das vorliegende Programm speichert bis zu 3 volle Disketten auf einer einzigen C60 Kassette ab, eine Diskettenseite in weniger als 12 min. Das Programm ist vollständig menugesteuert und erklärt sich selber. Die Sicherheitskopien bestehen aus 5 Blöcken zu je 36 kbyte. Sie sind nicht auf Kassette lauffähig, sondern müssen erst wieder auf eine leere Diskettenseite überspielt werden. Es empfiehlt sich hochwertige, leere Kassetten als Backupmedium zu benutzen oder die Baudrate zu reduzieren.

Variablenliste

a	Baudratenparameter, Menuindex
a$	Eingabevariable
bl	Block
fe	Fehlernummer
se	Sektor
sp	Spur
t	Bezugsspur

```
1 REM ************************************
2 REM *          B A C K U P             *
3 REM *   Copyright       Helmut Wenz    *
4 REM ************************************
10 ¦DISC:MODE 1:LOCATE 12,6:PRINT"B  A  C  K  U  P"
20 LOCATE 5,8:PRINT"Diskette und Kassette einlegen"
30 LOCATE 13,10:PRINT"Taste druecken"
40 CALL &BB06:CLS:CAT
50 FOR i=&A000 TO &A09B:READ a$:POKE i,VAL("&"+a$):NEXT
53 REM
55 REM  Diskettenroutine (Lesen und Schreiben)
57 REM
60 DATA 21,20,A0,CD,D4,BC,79,32,23,A0,22,21,A0,21,24,A0
70 DATA 4E,23,56,23,5E,21,00,A1,DF,21,A0,C9,00,00,00,00
73 REM
75 REM  Datenfeld fuer Diskettenroutine
77 REM
80 DATA 84,00,00,00,00,00,00,FF,00,00,00,00,00,00,00,00
83 REM
85 REM  Block-Leseroutine fuer Datacorder
87 REM
90 DATA 21,00,10,11,00,90,3E,16,CD,A1,BC,D8,32,27,A0,C9
93 REM
95 REM  Block-Schreibroutine fuer Datacorder
97 REM
100 DATA 21,00,10,11,00;90,3E,16,CD,9E,BC,D8,32,27,A0,C9
103 REM
105 REM  Schieberoutinen fuer einzelne Sektoren
```



```
107 REM
110 DATA 21,00,10,11,00,A1,01,00,02,ED,B0,C9,00,00,00,00
120 DATA 21,00,A1,11,00,10,01,00,02,ED,B0,C9,00,00,00,00
123 REM
125 REM  Einstellen der Baudrate
127 REM
130 DATA 21,5D,00,3E,0A,CD,68,BC,C9,00,00,00,00,00,00,00
133 REM
135 REM  Sektorvergleich
137 REM
140 DATA 21,00,00,11,00,A1,01,00,02,7E,EB,BE,20,08,23,13
150 DATA 0B,78,B1,C8,18,F3,3E,04,32,27,A0,C9
160 LOCATE 13,24:PRINT"Taste druecken!":CALL &BB06
163 REM
165 REM  Hauptmenu:  Laden, Vergleichen, Sichern
167 REM
170 CLS:LOCATE 5,8:PRINT"Von Kassette auf Diskette  (1)"
180 LOCATE 5,10:PRINT"Vergleich Kassette - Disk  (2)"
190 LOCATE 5,12:PRINT"Von Diskette auf Kassette  (3)"
200 a$=INKEY$:IF a$<"1" OR a$>"3" THEN 200
210 IF a$="3" THEN 370 ELSE a=VAL(a$)
220 CLS:LOCATE 5,10:PRINT"<PLAY>  und  <Enter> druecken!"
223 REM
225 REM  Programme:  Laden, Vergleichen
227 REM
230 CALL &BB06:CLS:FOR bl=1 TO 5:bu=&E:t=bl*8-8:CLS
240 LOCATE 10,8:PRINT "Block"bl:¦TAPE:CALL &A030:¦DISC
250 IF PEEK(&A027)<>255 THEN 550
260 FOR sp=t TO t+7:FOR se=0 TO 8
270 LOCATE 10,10:PRINT"Spur"sp,"Sektor"se
280 bu=bu+2:ON a GOSUB 300,330
283 REM
285 REM  Warten bis Diskettenmotor aus
287 REM
290 NEXT:NEXT:FOR I=0 TO 6000:NEXT:NEXT:GOTO 540
293 REM
295 REM  Unterprogramm:  Laden
297 REM
300 POKE &A052,bu:CALL &A050:POKE &A020,&85:POKE &A026,0
310 POKE &A025,sp:POKE &A024,se+PEEK(&A89F):CALL &A000
320 RETURN
323 REM
325 REM  Unterprogramm:  Vergleichen
327 REM
```

```
330 POKE &A020,&84:POKE &A026,0
340 POKE &A025,sp:POKE &A024,se+PEEK(&A89F):CALL &A000
350 POKE &A082,bu:CALL &A080
360 IF PEEK(&A027)<>255 THEN GOTO 550 ELSE RETURN
363 REM
365 REM  Programm:  Sichern
367 REM
370 a=110:CLS:LOCATE 7,8:PRINT"<REC> und <PLAY> druecken!"
380 LOCATE 9,10:PRINT"Schreibgeschwindigkeit"
390 LOCATE 9,12:PRINT"1300  Baud          (0)"
400 LOCATE 9,14:PRINT"2200  Baud          (1)"
410 LOCATE 9,16:PRINT"3000  Baud          (2)"
420 a$=INKEY$:IF a$<"0" OR a$>"2" THEN 420
430 IF a$="0" THEN a=255 ELSE IF a$="1" THEN a=150
440 CLS:POKE &A071,a:CALL &A070
450 FOR bl=1 TO 5:bu=&E:t=bl*8-8
460 FOR sp=t TO t+7:FOR se=0 TO 8
470 LOCATE 10,10:PRINT"Spur"sp,"Sektor"se:bu=bu+2
480 POKE &A020,&84:POKE &A026,0
490 POKE &A025,sp:POKE &A024,se+PEEK(&A89F):CALL &A000
500 POKE &A065,bu:CALL &A060
510 NEXT:NEXT:CLS:LOCATE 10,8:PRINT"Block"bl
513 REM
515 REM  Warten bis Diskettenmotor aus
517 REM
520 FOR i=0 TO 6000:NEXT i:|TAPE:CALL &A040:|DISC
530 IF PEEK(&A027)<>255 THEN GOTO 550 ELSE NEXT
540 PRINT CHR$(7):END
543 REM
545 REM  Ausgabe:  Fehlermeldung, Fehlernummer
547 REM
550 fe=PEEK(&A027):LOCATE 10,12:PRINT"F E H L E R "fe
```

Vorstellung: Lunar-Computer-Club

Der LCC ist am 1. 7. 1985 gegründet worden und beschäftigt sich hauptsächlich mit MSX-Computersystemen, MSX-Hardware und -Software, Bücher, Tips und Tricks, usw.; außerdem wird Clubmitgliederbetreuung groß geschrieben.

Mitglied im LCC kann jeder werden, auch Firmen sind willkommen. Zusammengefaßt bietet der LCC seinen Mitgliedern folgende Leistungen:

Händlerangebotslisten (Hard- und Software je nach Angebot)

Tauschlisten (LCC-Free-Soft-Programme)

Hard- u. Software-Marktlisten (über die Produkte, die der LCC preiswert erhält)

LCC Hard- u. Softwarelisten (eigene Hard- u. Softwareentwicklungen)

Clubzeitschrift (Clubbuch in Vorbereitung)

Tips und Tricks/Infos (Sammellisten)

Mitgliederlisten (Sammellisten)

Ordentliche Mitglieder haben einen Jahresbeitrag von 30,-- DM zu entrichten. Außerordentliche Mitglieder müssen für die Clubzeitschrift und für Listen, Tips und Tricks etc. eine je nach Umfang, Liste und Porto bestimmte Schutzgebühr entrichten. Kontaktadressen:

Lunar-Computer-Club
z. Hd. Herrn A. Potthoff
Am Wasserturm 3-5
4400 Münster
Tel.: 02 51 - 7 51 05

Lunar-Computer-Club
z. Hd. Herrn P. Brandt
Postfach 1542
4400 Münster
Tel.: 02 51 - 61 64 80
(mit Anrufbeantworter)

Basic-Befehlserweiterung für den Schneider CPC 464 um den Befehl ELLIPSE

Bei dieser Maschinenroutine handelt es sich um eine RSX-Erweiterung für den Schneider CPC 464, das den Basic-Befehlssatz um den Befehl ELLIPSE erweitert.

Der Basic-Lader kann nach einmaligem Aufruf gelöscht werden (vorher, wenn nicht schon geschehen, abspeichern!).
Hat man die Befehlserweiterung einmal im Speicher, so kann diese auch direkt als Binärfile mit
SAVE "ELLJPSE",-B,&A32D,&140
auf Disc oder Cassette gespeichert werden.

Vor dem erneuten Laden des Binärfiles ist der HIMEM durch MEMORY &A2FF neu zu setzen, nach dem Laden muß die Routine einmal durch CALL &A32D aufgerufen werden; danach befindet sie sich in der RSX-Erweiterung.

Der Befehlsaufruf vom Basic aus geschieht durch:
iELLIPSE,<X-ORIGIN>,<Y-ORIGIN>,<RADIUS(X)>,<RADIUS(Y)>,<WINKEL>,[<PEN>]
"i" (Balken über dem Klammeraffen) ist wie bei allen RSX-Erweiterungen beim Schneider CPC 464 dem Befehl voranzustellen, weiterhin muß unbedingt das Komma hinter dem Befehlswort eingegeben werden.

Die Routine erwartet mindestens 5 Parameter, die beiden Werte zur Bestimmung des Mittelpunks, die beiden Radien, sowie den Winkel um den die Ellipse gegenüber der normalen Lage [liegend, wenn r(x)>r(y),stehend, wenn r(y)>r(x)] sein soll.
Als sechster Parameter kann die PEN-Farbe eingegeben werden. Wird dieser Parameter angegeben, so stellt die Routine nach dem Zeichen der Ellipse, die alte PEN-Farbe wieder ein. Wird die PEN-Farbe nicht angegeben, so wird in der jeweils aktuellen Farbe gezeichnet.

Bei allen einzugebenden Parametern muß es sich um natürliche Zahlen handeln.

Die Drehung der Ellipse wird über die folgende Formel berechnet:

xl = x * cos (phi) + y * sin (phi)
yl = -x * sin (phi) + y * cos (phi)
mit: x=r(x)*sin(alpha)
y=r(y)*cos(alpha)

```
1 '*****************************************************************************
2 '*          RSX - ERWEITERUNG :     NACHLADBARER ELLIPSE - BEFEHL     *
3 '*                                                                    *
4 '* FORMAT :                                                          *
5 '*                                                                    *
6 '* |ELLIPSE,<X-ORI>,<Y-ORI>,<RADIUS(X)>,<RADIUS(Y)>,<WINKEL>,[<PEN>] *
7 '*                                                                    *
8 '*  WOLFGANG BOENIGK,  LANDRAT-VON-OSTMAN-STR. 3,  4500 OSNABRUECK    *
9 '*****************************************************************************
100 '
110 MEMORY &A2FF
120 adr=&A32D
130 READ a$
140 IF a$="ende" GOTO 210
150 a$="&"+a$
160 a=VAL(a$)
170 POKE adr,a
180 control=control+a
190 adr=adr+1
200 GOTO 130
210 IF control <> 38630 THEN PRINT "Fehler in DATA's !":END
220 CALL &A32D
230 '
240  DATA 01,36,A3,21,43,A3,C3,D1,BC,3B,A3,C3,47,A3,45,4C,4C
250  DATA 49,50,53,C5,00,00,00,00,00,F5,CD,E1,BB,67,F1,FE,05
260  DATA 28,0D,FE,06,C0,DD,7E,00,CD,DE,BB,DD,23,DD,23,7C,F5
270  DATA DD,56,09,DD,5E,08,DD,66,07,DD,6E,06,CD,C9,BB,DD,66
280  DATA 01,DD,6E,00,22,0A,A3,DD,66,03,DD,6E,02,22,05,A3,DD
```

```
290  DATA 66,05,DD,6E,04,7C,11,00,A3,CD,40,BD,2A,05,A3,7C,11
300  DATA 05,A3,CD,40,BD,2A,0A,A3,7C,11,0A,A3,CD,40,BD,3E,FF
310  DATA 32,F7,B8,01,00,00,C5,C5,E1,7C,11,23,A3,D5,CD,40,BD
320  DATA E1,11,28,A3,01,05,00,ED,B0,21,23,A3,E5,CD,88,BD,E1
330  DATA 11,00,A3,CD,61,BD,21,28,A3,E5,CD,8B,BD,E1,11,05,A3
340  DATA CD,61,BD,21,0A,A3,11,0F,A3,D5,01,05,00,ED,B0,E1,E5
350  DATA CD,8B,BD,E1,11,19,A3,01,05,00,ED,B0,21,0A,A3,11,14
360  DATA A3,D5,01,05,00,ED,B0,E1,E5,CD,88,BD,E1,11,1E,A3,01
370  DATA 05,00,ED,B0,21,0F,A3,11,23,A3,CD,61,BD,21,14,A3,11
380  DATA 28,A3,CD,61,BD,21,0F,A3,11,14,A3,CD,58,BD,21,19,A3
390  DATA E5,11,28,A3,CD,61,BD,21,1E,A3,E5,11,23,A3,CD,61,BD
400  DATA D1,E1,CD,5B,BD,21,0F,A3,CD,63,A4,E5,21,19,A3,CD,63
410  DATA A4,D1,CD,EA,BB,C1,03,21,68,01,AF,ED,42,C2,AA,A3,F1
420  DATA CD,DE,BB,C9,CD,46,BD,DD,CB,03,7E,C4,C7,BD,C9,ende
430 '
440 'BEISPIELPROGRAMM
450 '
460 MODE 1
470 MOVE 0,200 : DRAWR 640,0
480 MOVE 320,0 : DRAWR 0,400
490 |ELLIPSE,320,200,150,75,0
500 |ELLIPSE,320,200,150,75,45,2
510 |ELLIPSE,320,200,150,75,135
```

Assembler-Listing

```
A000              10                ;**************************************
A000              20                ;**     RSX - Erweiterung            **
A000              30                ;** Nachladbarer ELLIPSE - Befehl  **
A000              40                ;**     Wolfgang Boenigk             **
A000              50                ;**************************************
A000              60                ;
A300              70          ORG   &A300
A300              80  LOGEXT  EQU   &BCD1
A300              90  GETPEN  EQU   &BBE1
A300             100  SETPEN  EQU   &BBDE
A300             110  ORIGIN  EQU   &BBC9
A300             120  INTFLO  EQU   &BD40 ;  Integer nach Fliesskomma
A300             130  SINUS   EQU   &BD88
A300             140  COS     EQU   &BD8B
A300             150  MULT    EQU   &BD61 ;     Fliesskommaoperation
A300             160  ADDIER  EQU   &BD58 ;     Fliesskommaoperation
A300             170  SUBTRA  EQU   &BD5B ;     Fliesskommaoperation
A300             180  PLOT    EQU   &BBEA
A300             190  FLOINT  EQU   &BD46 ;  Fliesskomma nach Integer
A300             200  CHGSGN  EQU   &BDC7 ;  Vorzeichwechsel Integer
A300             210  RX      DEFS  5 ;       Radius von x
A305             220  RY      DEFS  5 ;       Radius von y
A30A             230  PHI     DEFS  5
A30F             240  ZW1     DEFS  5
A314             250  ZW2     DEFS  5
A319             260  ZW3     DEFS  5
A31E             270  ZW4     DEFS  5
A323             280  X       DEFS  5
A328             290  Y       DEFS  5
A32D             300                ;
A32D 010000      310          LD    BC,RSX
A330 210000      320          LD    HL,KERNAL
A333 C3D1BC      330          JP    LOGEXT
```



```
**** Zeile  310  : RSX=&A336
A336 0000      340  RSX     DEFW TABLE
A338 C30000    350          JP   ELLIPS
**** Zeile  340  : TABLE=&A33B
A33B 454C4C495053
                360  TABLE   DEFM "ELLIPS"
A341 C5        370          DEFB &C5 ;          "E"+&80
A342 00        380          DEFB 0
**** Zeile  320  : KERNAL=&A343
A343           390  KERNAL  DEFS 4
**** Zeile  350  : ELLIPS=&A347
A347 F5        400  ELLIPS  PUSH AF
A348 CDE1BB    410          CALL GETPEN
A34B 67        420          LD   H,A
A34C F1        430          POP  AF
A34D FE05      440          CP   5
A34F 28FE      450          JR   Z,DOELLI
A351 FE06      460          CP   6
A353 C0        470          RET  NZ
A354 DD7E00    480          LD   A,(IX+0) ;     wenn zusaetzlich
A357 CDDEBB    490          CALL SETPEN ;       Pen angegeben
A35A DD23      500          INC  IX
A35C DD23      510          INC  IX
**** Zeile  450  : DOELLI=&A35E
A35E 7C        520  DOELLI  LD   A,H
A35F F5        530          PUSH AF
A360 DD5609    540          LD   D,(IX+9)
A363 DD5E08    550          LD   E,(IX+8) ;     X-Position von Origin
A366 DD6607    560          LD   H,(IX+7)
A369 DD6E06    570          LD   L,(IX+6) ;     Y-Position von Origin
A36C CDC9BB    580          CALL ORIGIN
A36F DD6601    590          LD   H,(IX+1) ;     Winkel
A372 DD6E00    600          LD   L,(IX+0)
A375 220AA3    610          LD   (PHI),HL
A378 DD6603    620          LD   H,(IX+3) ;     Radius (y)
A37B DD6E02    630          LD   L,(IX+2)
A37E 2205A3    640          LD   (RY),HL
A381 DD6605    650          LD   H,(IX+5) ;     Radius (x)
A384 DD6E04    660          LD   L,(IX+4)
A387 7C        670          LD   A,H
A388 1100A3    680          LD   DE,RX
A38B CD40BD    690          CALL INTFLO
A38E 2A05A3    700          LD   HL,(RY)
A391 7C        710          LD   A,H
A392 1105A3    720          LD   DE,RY
A395 CD40BD    730          CALL INTFLO
A398 2A0AA3    740          LD   HL,(PHI)
A39B 7C        750          LD   A,H
A39C 110AA3    760          LD   DE,PHI
A39F CD40BD    770          CALL INTFLO
A3A2 3EFF      780          LD   A,&FF
A3A4 32F7B8    790          LD   (&B8F7),A ;    -> DEG
A3A7 010000    800          LD   BC,0 ;         Zaehler
A3AA C5        810  LOOP    PUSH BC
A3AB C5        820          PUSH BC
A3AC E1        830          POP  HL
A3AD 7C        840          LD   A,H
A3AE.1123A3    850          LD   DE,X
A3B1 D5        860          PUSH DE
A3B2 CD40BD    870          CALL INTFLO ;       Zaehler nach FLO
A3B5 E1        880          POP  HL
```

```
A3B6 1128A3   890          LD   DE,Y
A3B9 010500   900          LD   BC,5
A3BC EDB0     910          LDIR  ;          Zaehler v.X->Y kopieren
A3BE 2123A3   920          LD   HL,X
A3C1 E5       930          PUSH HL
A3C2 CD88BD   940          CALL SINUS ;     sin(z)
A3C5 E1       950          POP  HL
A3C6 1100A3   960          LD   DE,RX
A3C9 CD61BD   970          CALL MULT ;      r(x)*sin(z)
A3CC 2128A3   980          LD   HL,Y
A3CF E5       990          PUSH HL
A3D0 CD8BBD   1000         CALL COS ;       cos(z)
A3D3 E1       1010         POP  HL
A3D4 1105A3   1020         LD   DE,RY
A3D7 CD61BD   1030         CALL MULT ;      r(y)*cos(z)
A3DA 210AA3   1040         LD   HL,PHI
A3DD 110FA3   1050         LD   DE,ZW1
A3E0 D5       1060         PUSH DE
A3E1 010500   1070         LD   BC,5
A3E4 EDB0     1080         LDIR  ;          phi nach ZW1
A3E6 E1       1090         POP  HL
A3E7 E5       1100         PUSH HL
A3E8 CD8BBD   1110         CALL COS ;       cos(phi)
A3EB E1       1120         POP  HL
A3EC 1119A3   1130         LD   DE,ZW3
A3EF 010500   1140         LD   BC,5
A3F2 EDB0     1150         LDIR  ;          cos(phi) nach ZW3 kopieren
A3F4 210AA3   1160         LD   HL,PHI
A3F7 1114A3   1170         LD   DE,ZW2
A3FA D5       1180         PUSH DE
A3FB 010500   1190         LD   BC,5
A3FE EDB0     1200         LDIR  ;          phi nach ZW2
A400 E1       1210         POP  HL
A401 E5       1220         PUSH HL
A402 CD88BD   1230         CALL SINUS ;     sin(phi)
A405 E1       1240         POP  HL
A406 111EA3   1250         LD   DE,ZW4
A409 010500   1260         LD   BC,5
A40C EDB0     1270         LDIR  ;          sin(phi) nach ZW4 kopieren
A40E 210FA3   1280         LD   HL,ZW1
A411 1123A3   1290         LD   DE,X
A414 CD61BD   1300         CALL MULT ;      cos(phi)*x
A417 2114A3   1310          LD   HL,ZW2
A41A 1128A3   1320          LD   DE,Y
A41D CD61BD   1330          CALL MULT ;      sin(phi)*y
A420 210FA3   1340          LD   HL,ZW1
A423 1114A3   1350          LD   DE,ZW2
A426 CD58BD   1360          CALL ADDIER ;    cos(phi)*x+sin(phi)*y
A429 2119A3   1370          LD   HL,ZW3
A42C E5       1380          PUSH HL
A42D 1128A3   1390          LD   DE,Y
A430 CD61BD   1400          CALL MULT ;      cos(phi)*y
A433 211EA3   1410          LD   HL,ZW4
A436 E5       1420          PUSH HL
A437 1123A3   1430          LD   DE,X
A43A CD61BD   1440          CALL MULT ;      sin(phi)*x
A43D D1       1450          POP  DE
A43E E1       1460          POP  HL
A43F CD5BBD   1470          CALL SUBTRA ;    -sin(phi)*x+cos(phi)*y
A442 210FA3   1480          LD   HL,ZW1
A445 CD0000   1490          CALL SFI
```



```
A448 E5       1500          PUSH HL
A449 2119A3   1510          LD   HL,ZW3
A44C CD0000   1520          CALL SFI
A44F D1       1530          POP  DE
A450 CDEABB   1540          CALL PLOT
A453 C1       1550          POP  BC
A454 03       1560          INC  BC
A455 216801   1570          LD   HL,360
A458 AF       1580          XOR  A ;              Carry loeschen
A459 ED42     1590          SBC  HL,BC
A45B C2AAA3   1600          JP   NZ,LOOP
A45E F1       1610          POP  AF
A45F CDDEBB   1620          CALL SETPEN
A462 C9       1630          RET
**** Zeile    1490  : SFI=&A463
**** Zeile    1520  : SFI=&A463
A463 CD46BD   1640  SFI       CALL FLOINT
A466 DDCB037E 1650            BIT  7,(IX+3) ;     Zahl negativ ?
A46A C4C7BD   1660            CALL NZ,CHGSGN ;    wenn ja, dann Vorzeichen
A46D          1670                  ;             von Integerzahl wechseln
A46D C9       1680            RET

Programm :BASIC - Erweiterung um den Befehl ELLIPSE
Start : &A300     Ende : &A46D
Laenge : 016E
```

Dieses Programm ist auf dem CPC-464 und CPC-664 lauffähig

Directory-Ausgabe auf dem Drucker

Beim Schneider CPC-464 und CPC-664 stehen zwei Befehle zur Ausgabe des Disketten-Directory zur Verfügung: !DIR und CAT. Doch beide funktionieren nur mit Bildschirmausgabe, nicht jedoch auf dem Drucker. Dieses Maschinenprogramm für beide Schneider-Computer erweitert das Locomotive-Basic um zwei neue RSX-Befehle, mit denen das Disketten-Inhaltsverzeichnis auf dem Drucker ausgegeben werden kann.

Bei der Programmeingabe gibt es zwei Wege: Sie können entweder den Quellcode des Maschinenprogramms abtippen und assemblieren lassen oder den Basic-Loader eingeben.

Falls Sie einen CPC-664 besitzen und den Quellcode eingeben wollen, beachten Sie bitte die Zeilen 280 bis 300: Dort müssen, die im Kommentarfeld angegebenen Konstanten-Referenzen für DSKOFF, GETMEM und RELMEM statt der originalen, links davon stehenden Werte, eingegeben werden.

Beim Basic-Loader ist dies viel einfacher: Nach dem Programmstart wird eine kurze Anleitung ausgegeben und der Maschinencode eingelesen. Danach werden Sie gefragt, welchen der beiden Computer Sie besitzen. Dementsprechend werden die Sprungadressen an die unterschiedlichen ROM-Versionen des Basic-Interpreters angepaßt.

Der Maschinencode ist zwar nicht frei relokatierbar, das Loader-Programm paßt aber alle internen absoluten Adressen an. Somit wird das Maschinenprogramm immer direkt unter HIMEM gelegt und nimmt den geringstmöglichen Speicherplatz weg.

Sobald die RSX-Erweiterung geladen ist, verfügen Sie über zwei neue Befehle:

!PDIR – entspricht dem normalen !DIR-Befehl

!PCAT – entspricht dem normalen Disketten-CAT

Achten Sie aber darauf, daß der Drucker empfangsbereit ist, denn der Computer wartet auf jeden Fall, bis der Katalog ausgegeben werden kann.

Für Programmierer soll hier noch kurz die Funktionsweise des Programms erläutert werden: Wie inzwischen allgemein bekannt ist, werden die Hauptfunktionen des Betriebssystems in den CPCs über Vektoren gesteuert, die im RAM stehen. Dadurch können diese Sprungvektoren verändert werden. Beide Katalogbefehle, als !DIR und CAT verwenden, sobald eine Diskettenstation angeschlossen ist, zur Ausgabe des Disketteninhalts die Betriebssystem-Routine TXT OUTPUT, mit der jeweils ein Buchstabe auf den Bildschirm gebracht werden kann. Das Programm verändert nun diesen Zeiger auf ein eigenes Unterprogramm hin, das statt des Bildschirms den Drucker ansteuert. Nach der vollständigen Ausgabe des Katalogs werden die Vektoren wieder auf die ursprünglichen Werte zurückgesetzt.

Bleibt nur noch das Problem, die Befehle CAT und !DIR im Floppy-ROM aufzurufen: Bei CAT ist dies relativ einfach, da hier wie bei TXT OUTPUT ein Sprungvektor im Hauptspeicher verwendet wird. Somit kann das Unterprogramm mit dem Z80-Befehl CALL aufgerufen werden. CAT hat aber die „unangenehme“ Eigenschaft, den Katalog nicht nur auf den Bildschirm zu schreiben, sondern auch eine Kopie davon in den Speicher. Dazu muß der Routine ein Zeiger auf einen 2K-Puffer übergeben werden. Diesen zu suchen, vor Basic zu schützen und nachher wieder freizugeben, würde eine Menge Programmieraufwand verursachen; darum verwendet das

Programm Hilfsroutinen im Basic-ROM, was aber zu neuen Problemen führt:
Da das Programm auch mit dem CPC-664 funktionieren soll, dort aber das komplette Basic-ROM überarbeitet wurde, müssen alle drei Einsprünge in den Basic-Interpreter in der 664-Version umgeändert werden – der Basic-Loader macht dies automatisch.

Der !DIR-Befehl legt keinen Pufferspeicher des Katalogs an, ist dafür aber auch nicht einfach mit CALL aufzurufen. Das Betriebssystem verwaltet solche RSX-Befehlserweiterungen mittels miteinander verknüpfter Sprungtabellen. Die RSXen können dann anhand ihres Namens mit der Betriebssystem-Routine KL FIND COMMAND in allen Speicherbereichen gesucht werden.

Als Ergebnis dieser Suche befindet sich die RSX-Routinenadresse im HL-Register, der ROM-Status im C-Register. Diese Registerwahl ist recht geschickt gemacht, denn genau diese Register werden auch von KL FAR PCHL benötigt, das jeden beliebigen Sprung in den RAM oder eines der 252 Erweiterungs-ROMs durchführen kann. Auch hier wird nach erfolgter Druckerausgabe der OUTPUT-Vektor wiederhergestellt, und das Programm kehrt zum Basic zurück.

Martin Kotulla

```
100 ' ****************************
110 ' *                          *
120 ' *    Druckerausgabe des    *
130 ' *    Diskettenkatalogs     *
140 ' *                          *
150 ' ****************************
160 '
170 DEF FNmsb(a)=255 AND INT(a/256)
180 DEF FNlsb(a)=255 AND UNT(a)
190 MODE 1:INK 0,1:INK 1,24:INK 2,6,16:INK 3,21
200 PAPER 0:PEN 3:BORDER 2:SPEED INK 30,20
210 LOCATE 1,1:PRINT STRING$(40,210);
220 LOCATE 1,2
230 PRINT CHR$(24)+"    DISKETTENKATALOG AUF DEM DRUCKER     "+CHR$(24)
240 LOCATE 10,5:PRINT CHR$(164)+" Martin Kotulla 1985"
250 PRINT STRING$(40,210)
260 PEN 1:PRINT " Dieses Maschinenprogramm erweitert den"
270 PRINT:PRINT " Befehlssatz des Computers um zwei"
280 PRINT:PRINT " RSX-Kommandos, mit denen das Inhalts-"
290 PRINT:PRINT " verzeichnis der Diskette auf dem"
300 PRINT:PRINT " Drucker ausgegeben werden kann:"
310 PRINT STRING$(40,210):PRINT
320 PEN 2:PRINT TAB(9);" |PCAT";:PEN 3:PRINT " entspricht CAT"
330 PEN 2:PRINT TAB(9);" |PDIR";:PEN 3:PRINT " entspricht |DIR":PRINT
340 FOR i=1 TO 122:READ a:sum=sum+a:NEXT i
350 IF sum=14976 THEN 380
360 MODE 2:INK 1,0:INK 0,10:BORDER 10:PRINT CHR$(7)
370 PRINT "* Fehler in den DATA-Zeilen!":PRINT:LIST 570-640
380 RESTORE 660:FOR i=1 TO 14:READ a:sum2=sum2+a:NEXT i
390 IF sum2=681 THEN 420 ' Zweiter Datenblock auch korrekt!
400 MODE 2:INK 1,0:INK 0,10:BORDER 10:PRINT CHR$(7)
410 PRINT "* Fehler in den Relokatier-Daten!":PRINT:LIST 660
420 RESTORE:MEMORY HIMEM-124:start=HIMEM+1
430 FOR i=start TO start+121:READ a:POKE i,a:NEXT i
440 FOR i=1 TO 14:READ a:a=a+start
450   value=PEEK(a)+PEEK(a+1)*256-40960+start
460   POKE a,FNlsb(value):POKE a+1,FNmsb(value)
470 NEXT i
480 PRINT " Computertyp: 1> CPC-464  2> CPC-664 ";:PEN 2:PRINT CHR$(143);
490 in$=INKEY$:IF in$="" THEN 490
500 IF in$<>"1" AND in$<>"2" THEN PRINT CHR$(7);:GOTO 490
510 IF in$="1" THEN 550 ' Beim CPC-464 ist keine Anpassung notwendig
520 POKE start+44,&3:POKE start+45,&D3 '  DSKOFF an den CPC-664 anpassen
530 POKE start+47,&2A:POKE start+48,&F7 ' GETMEM an den CPC-664 anpassen
540 POKE start+53,&5D:POKE start+54,&F7 ' RELMEM an den CPC-664 anpassen
```



```
550 PEN 1:MODE 1:BORDER 1:CALL start+&1B ' RSX-Erweiterung einbinden
560 ' Maschinencode-Daten ***********************************************
570 DATA &FC,&A6,&0A,&A0,&CF,&00,&94,&C3,&74,&A0,&12,&A0,&C3,&25,&A0,&C3
580 DATA &3B,&A0,&50,&43,&41,&D4,&50,&44,&49,&D2,&00,&01,&0A,&A0,&21,&00
590 DATA &A0,&CD,&D1,&BC,&C9,&CD,&51,&A0,&CD,&00,&B9,&CD,&AD,&D2,&CD,&37
600 DATA &F6,&CD,&9B,&BC,&CD,&71,&F6,&CD,&68,&A0,&C9,&CD,&51,&A0,&21,&4E
610 DATA &A0,&CD,&D4,&BC,&30,&04,&97,&CD,&1B,&00,&CD,&68,&A0,&C9,&44,&49
620 DATA &D2,&21,&5A,&BB,&11,&04,&A0,&01,&03,&00,&ED,&B0,&21,&07,&A0,&11
630 DATA &5A,&BB,&01,&03,&00,&ED,&B0,&C9,&21,&04,&A0,&11,&5A,&BB,&01,&03
640 DATA &00,&ED,&B0,&C9,&CD,&2B,&BD,&30,&FB,&C9
650 ' Daten zum Relokatieren des Maschinencodes ************************
660 DATA 8,10,13,16,28,31,38,56,60,63,75,85,93,105
```

Assemblerlisting

```
100 ' ; ***************************************************************
110 ' ; *                                                             *
120 ' ; *   RSX-Erweiterung zur Druckerausgabe des Diskettenkatalogs   *
130 ' ; *                                                             *
140 ' ; ***************************************************************
150 '
160 '        ORG  &A000       ; Der Basic-Loader ist relokatibel!
170 '
180 ' UROMON EQU  &B900       ; KL UPPER ROM ENABLE: Basic-ROM einschalten
190 ' TXTOUT EQU  &BB5A       ; TXT OUTPUT: Bildschirmanzeige eines Zeichens
200 ' PRTCHR EQU  &BD2B       ; MC PRINT CHAR: Druckerausgabe eines Zeichens
210 ' DSKCAT EQU  &BC9B       ; DISK CATALOG: Disketten-Katalog auf Bildschirm
220 ' FINDCM EQU  &BCD4       ; KL FIND COMMAND: RSX-Adresse suchen
230 ' PCHL   EQU  &001B       ; KL FAR PCHL: Floppy-ROM-Routinen aufrufen
240 ' LOGEXT EQU  &BCD1       ; KL LOG EXTERNAL: RSX-Befehle einbinden
250 '
260 ' ; ****** Routinen im Basic-Interpreter des CPC-464/664 *****************
270 '
280 ' DSKOFF EQU  &D2AD       ; 664: DSKOFF EQU &D303  Disketten-I/O abbrechen
290 ' GETMEM EQU  &F637       ; 664: GETMEM EQU &F72A  Ausgabepuffer anlegen
300 ' RELMEM EQU  &F671       ; 664: RELMEM EQU &F75D  Ausgabepuffer freigeben
310 '
320 ' ; ****** Sprungvektor und Hilfsspeicher ******************************
330 '
340 ' MEMORY DEFS 4           ; Hilfsspeicher zur RSX-Verwaltung
350 ' TXTBUF DEFS 3           ; Buffer fuer den Vektor von TXT OUTPUT
360 ' PRTOUT JP   PRINTR      ; Neuer Inhalt fuer TXT OUTPUT waehrend PDIR/PCAT
370 '
380 ' RSXTAB DEFW NAMTAB      ; Zeiger auf RSX-Namenstabelle
390 '        JP   CAT         ; Sprung nach PCAT
400 '        JP   PRTDIR      ; Sprung nach PDIR
410 '
420 ' NAMTAB DEFB &50,&43,&41,&D4
430 '        DEFB &50,&44,&49,&D2
440 '        DEFB &00
450 '
460 ' INIT   LD   BC,RSXTAB   ; BC:=Zeiger auf Sprungtabelle
470 '        LD   HL,MEMORY   ; HL:=Zeiger auf Hilfsspeicher fuer Kernel-ROM
480 '        CALL LOGEXT      ; RSX-Erweiterungen einbinden
```

```
490 '       RET             ; Ruecksprung nach Basic
500 '
510 ' ; ****** Hauptprogramm: PCAT und PDIR **********************************
520 '
530 ' CAT   CALL PATCH      ; TXT OUTPUT nach MC PRINT CHAR umbiegen
540 '       CALL UROMON     ; Basic-ROM von &C000 bis &FFFF einschalten
550 '       CALL DSKOFF     ; Disketten-I/O abbrechen
560 '       CALL GETMEM     ; Speicher fuer Disc-Katalog reservieren
570 '       CALL DSKCAT     ; Katalog auf dem Drucker ausgeben
580 '       CALL RELMEM     ; Ausgabepuffer wieder freigeben
590 '       CALL RESTOR     ; TXT OUTPUT wieder auf Originalwert bringen
600 '       RET             ; Ruecksprung nach Basic
610 '
620 ' PRTDIR CALL PATCH     ; TXT OUTPUT auf MC PRINT CHAR umbiegen
630 '       LD   HL,NAMDIR  ; Zeiger auf den RSX-Namen "DIR"
640 '       CALL FINDCM     ; KL FIND COMMAND - sucht RSX-Adresse und ROM-Nr.
650 '       JR   NC,RETURN  ; Carry geloescht: RSX nicht gefunden
660 '       SUB  A          ; Akku loeschen: DIR ohne Parameter aufrufen
670 '       CALL PCHL       ; KL FAR PCHL: DIR im Floppy-ROM ansprechen
680 '
690 ' RETURN CALL RESTOR    ; TXT OUTPUT wieder auf Bildschirmausgabe bringen
700 '       RET             ; Ruecksprung nach Basic
710 '
720 ' NAMDIR DEFB &44,&49,&D2 ; "DI", "R"+&80: RSX-Name fuer Katalogausgabe
730 '
740 ' PATCH  LD   HL,TXTOUT ; Inhalt von TXT OUTPUT in TXTBUF sichern
750 '       LD   DE,TXTBUF
760 '       LD   BC,3
770 '       LDIR
780 '       LD   HL,PRTOUT  ; Sprung zur Druckerausgabe nach TXT OUTPUT
790 '       LD   DE,TXTOUT
800 '       LD   BC,3
810 '       LDIR
820 '       RET             ; Ruecksprung zum Hauptprogramm
830 '
840 ' RESTOR LD   HL,TXTBUF ; Den alten Inhalt von TXT OUTPUT wiederherstellen
850 '       LD   DE,TXTOUT
860 '       LD   BC,3
870 '       LDIR
880 '       RET
890 '
900 ' PRINTR CALL PRTCHR    ; MC PRINT CHAR: Zeichen auf dem Drucker ausgeben
910 '       JR   NC,PRINTR  ; So lange versuchen, bis es klappt
920 '       RET             ; Ruecksprung zum aufrufenden Programm
930 '       END             ; ************************************************
```

Beispiel: |PDIR-Befehl

```
Drive A: user  0

MOVCPM  .COM    PIP     .COM    SUBMIT  .COM    XSUB    .COM    ED      .COM
ASM     .COM    DDT     .COM    LOAD    .COM    STAT    .COM    DUMP    .COM
DUMP    .ASM    AMSDOS  .COM    FILECOPY.COM    SYSGEN  .COM    BOOTGEN .COM
DISCCOPY.COM    DISCCHK .COM    SETUP   .COM    FORMAT  .COM    CSAVE   .COM
CLOAD   .COM            85K free
```


Beispiel: |PCAT-Befehl

```
Drive A: user  0

AMSDOS  .COM   1K   DISCCHK .COM   3K   FORMAT  .COM   3K   SUBMIT  .COM   2K
ASM     .COM   8K   DISCCOPY.COM   3K   LOAD    .COM   2K   SYSGEN  .COM   2K
BOOTGEN .COM   2K   DUMP    .ASM   5K   MOVCPM  .COM  10K   XSUB    .COM   1K
CLOAD   .COM   2K   DUMP    .COM   1K   PIP     .COM   8K
CSAVE   .COM   2K   ED      .COM   7K   SETUP   .COM   8K
DDT     .COM   5K   FILECOPY.COM   3K   STAT    .COM   6K

 85K free
```

MSX

Zeichensatz-Konverter für Epson Drucker

MSX

Nachdem wir in der CPU Nr. 3 bereits ein Programm für MSX-Drucker veröffentlicht haben, folgt nun ein Programm für Epson-Drucker.

MSX-Computer sind mit einer parallelen Schnittstelle ausgestattet und können jeden Drukker mit Parallelschnittstelle steuern.
Viele Drucker können jedoch nicht den kompletten 254 Zeichen umfassenden MSX-Zeichensatz darstellen, so daß der Benutzer gezwungen ist, einen speziellen MSX-Drucker zu kaufen, auch wenn bereits ein Drucker vorhanden ist.
Der MSX-Zeichensatz-Konverter schafft dem Abhilfe, denn mit Hilfe dieses Programmes kann der komplette MSX-Zeichensatz auf jedem Epson-Drucker ausgedruckt werden.
Das Programm liest den kompletten MSX-Zeichensatz aus dem Videoram, wandelt diesen in für den Drucker verständliche Grafik um und speichert diesen umgewandelten Zeichensatz in einem Textvariablenfeld ab.
Dann werden die auszudrukkenden Dateien von Cassette gelesen und abgearbeitet. An den Drucker wird jetzt nicht mehr der ASCII-Code des Zeichens übertragen, sondern die umgewandelte Grafik des entsprechenden Zeichens. Somit kann der Drucker auch die Grafikzeichen ausgeben.
Das Programm berücksichtigt auch die Grafikzeichen, die durch 2 Bytes dargestellt werden. Somit werden alle 254 Zeichen für den Drucker darstellbar.

Doch nun zum Programmablauf:
MSX-Zeichensatz-Konverter können sämtliche im ASCII-Format gespeicherten Dateien von Cassette ausgeben. Um eine Datei ausgeben zu lassen, muß diese zunächst auf Cassette gespeichert werden. Dies geht im Normalfall vom Programm her mit einem OPEN-Befehl und anschließend mit PRINT #.
Um ein Listing mit richtigem MSX-Zeichensatz auszudrukken, muß das Programm mit SAVE "CAS:Name" abgespeichert werden, wobei Name jeder beliebige Programmname sein darf, da der Zeichensatz-Konverter den Programmnamen nicht abfragt.
Nachdem Sie Ihre Datei (oder Programm) auf Cassette gespeichert haben, laden Sie den Zeichensatz-Konverter ein und starten diesen mittels der Funktionstaste F5.
Auf dem Bildschirm sehen Sie jetzt den Schriftzug:

Zeichensatz wird umgewandelt bitte warten
Darunter erscheint die Schrift : Zeichen-> ### und Sie sehen aufwärtszählende Nummern. Dies gibt an, welches Zeichen gerade umgewandelt wird. Die Umwandlung dauert ca. 4 Minuten, und sobald Zeichen 255 umgewandelt ist, werden Sie aufgefordert, das Band zurückzuspulen, wobei der Recordermotor eingeschaltet wird. Wenn das Band zurückgespult ist, drücken Sie die <RETURN>-Taste und Sie werden aufgefordert, die Wiedergabe zu starten.
Sobald der Rechner jetzt eine ASCII-Datei gefunden hat, gibt er die Meldung „Druckvorgang ist eingeleitet" aus. Die Datei wird jetzt Zeile für Zeile eingelesen, abgearbeitet und an den Drucker ausgegeben, wobei die zur Zeit aktuelle Zeile der Datei im unteren Drittel des Bildschirmes ausgegeben wird.
Nachdem die Datei komplett ausgedruckt wurde, erscheint die Frage nach weiteren Ausdrucken. Wenn Sie noch weitere Dateien ausdrucken wollen, so antworten Sie mit „J", und es geht weiter mit der Aufforderung, das Band zurückzuspulen. Somit sparen Sie sich die Zeit der Zeichensatzumwandlung.
Wenn Sie keine Dateien mehr auszudrucken haben, so antworten Sie mit „N", um das Programm zu beenden.

Vorsicht

Wenn Sie den Zeichensatz-Konverter mittels <CTR>+<STOP> abbrechen, so muß unbedingt der Befehl CLEAR 200 gegeben werden, da der Zeichensatz-Konverter einen Bereich von 25 000 Bytes für Textfelder reserviert.
Wird CLEAR 200 nicht gegeben, so kommt es beim Einlesen des nächsten größeren Programmes zur Fehlermeldung „Out of Memory".
Wird das Programm jedoch mit „N" beendet, muß der Befehl CLEAR 200 nicht gegeben werden.
Die Speicherbelegung von 25 000 Bytes bleibt jedoch nur so lange bestehen, bis die <RESET>-Taste gedrückt wird.

An alle potentiellen MSX-Programmautoren!

Bei unserer Leserschaft besteht eine große Nachfrage nach MSX-Programmen. – Der Tronic-Verlag bezahlt 120,– DM pro abgedruckte Seite!

Die Formate der einzelnen Befehle:

Renew	
Print at spalte, zeile	Es sind Werte bis 39 bzw. 24 erlaubt
Volume wert	Es sind Werte bis 255 erlaubt
Fwave attack, sustain, wave	Es sind jeweils Werte bis 255 erlaubt. Dasselbe gilt für Swave und Twave
Fpulse low, high	Es müssen low- und highbyte angegeben werden. Dasselbe gilt für Spulse und Tpulse
Fplay low, high	Es müssen low- und highbyte der Freq. angegeben werden. Dasselbe für Splay und Tplay
System Cls	
Screen x, y	Der x-Wert ist für die Rand-, der y-Wert für den Bildschirm
Ink x	Der x-Wert bestimmt die Schriftfarbe
Repeat	
Unrepeat	
Execute „string“	Die Zeichenkette kann ein BASIC-Befehl sein oder eine Funktion enthalten.
Sreset	
Speed x	Wobei x die Geschwindigkeit angibt.
Susr low, high	Es müssen low- und highbyte der Adresse angegeben werden
Res x	Es sind Werte von 0-255 erlaubt
Multicol x, z	Es sind jeweils Werte bis 255 erlaubt.
Keybeep	
Beepoff	
Norm	
Multioff	

```
100 '●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●
110 '●                                ●
120 '●            M S X               ●
130 '●    Zeichensatz Konverter       ●
140 '●      für EPSON Drucker         ●
150 '●»«»«»«»«»«»«»«»«»«»«»«»«»«»«»«»●
160 '●                                ●
170 '●  (c) 1985 by Volker Becker     ●
180 '●                                ●
190 '●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●
200 '
210 '
220 CLEAR 25000
230 DEFINT I,J,K,L
240 DEFSTR P,X,Y,Z
250 DIM Y(8),Z(255)
260 P=CHR$(27)+"*"+CHR$(4)+CHR$(8)+CHR$(0)
270 KEY OFF:CLS:SCREEN 0:COLOR 14,1,1
280 LOCATE 6,0:PRINT "MSX Zeichensatz Konverter"
290 LOCATE 6,1:PRINT "------------------------"
300 LOCATE 6,3:PRINT "(c) 1985 by Volker Becker"
310 LOCATE 0,5:PRINT STRING$ (37,195)
320 LOCATE 7,7:PRINT "Zeichensatz umwandeln"
330 LOCATE 11,8:PRINT "bitte  warten"
340 LOCATE 11,10:PRINT "Zeichen->"
350 LOCATE 0,12:PRINT STRING$ (37,195)
360 FOR I=0 TO 255:LOCATE 21,10:PRINT USING "###";I:FOR K=0 TO 7:Y(K+1)=BIN$(VPE
EK(&H800+8*I+K))
370 IF LEN(Y(K+1))=8 THEN 390
380 FOR J=1+LEN(Y(K+1)) TO 8:Y(K+1)="0"+Y(K+1):NEXT
390 NEXT
400 FOR K=1 TO 8:X="":FOR J=1 TO 8:X=X+MID$(Y(J),K,1):NEXT J:Z(I)=Z(I)+CHR$(VAL(
"&B"+X)):NEXT K,I
410 MID$(Z(121),4,1)=MID$(Z(121),3,1):MID$(Z(152),3,1)=MID$(Z(152),2,1):MID$(Z(1
66),1,1)=CHR$(8)
420 FOR I=6 TO 11:LOCATE 0,I:PRINT STRING$(37,32):NEXT
430 LOCATE 9,7:PRINT "Spule Band zurück"
440 LOCATE 19,8:PRINT "┌──────┐"
450 LOCATE 8,9:PRINT "Dann drücke|RETURN|"
460 LOCATE 19,10:PRINT "└──────┘":BEEP:MOTOR ON
470 IF INKEY$=CHR$(13) THEN 480 ELSE 470
480 MOTOR OFF
490 LOCATE 9,7:PRINT "Starte Wiedergabe":BEEP
500 IF INKEY$=CHR$(13) THEN 510 ELSE 500
```



```
510 FOR I=6 TO 11:LOCATE 0,I:PRINT STRING$(37,32):NEXT
520 LOCATE 11,7
530 OPEN "CAS:" FOR INPUT AS #1
540 LOCATE 4,9:PRINT "Druckvorgang ist eingeleitet"
550 IF EOF(1) THEN 630
560 LINE INPUT #1,X
570 FOR I=14 TO 21:LOCATE 0,I:PRINT STRING$(37,32):NEXT
580 LOCATE 0,14:PRINT X
590 FOR I=1 TO LEN(X):J=ASC(MID$(X,I,1)):IF J>31 THEN 610
600 I=I+1:J=ASC(MID$(X,I,1))-64
610 LPRINT P;Z(J);:NEXT:LPRINT
620 GOTO 550
630 FOR I=14 TO 21:LOCATE 0,I:PRINT STRING$(37,32):NEXT
640 CLOSE
650 FOR I=6 TO 11:LOCATE 0,I:PRINT STRING$(37,32):NEXT
660 LOCATE 8,7:PRINT "Weitere Ausdrucke ?"
670 LOCATE 15,9:PRINT "[J/N]"
680 X=INKEY$
690 IF X="j" OR X="J" THEN 420
700 IF X="n" OR X="N" THEN 720
710 GOTO 680
720 CLEAR 200:CLS:COLOR 14,4,4:END
```

LADEN und SAVEN des BILDSCHIRMSPEICHERS auf Datasette!

MSX-Programm: „SCREENSAVE“

Bilder direkt aus dem Video RAM auf Kassette speichern ist eine schöne Sache. Bei den MSX Computern geht dies jedoch leider nur mit Diskette oder Quick Disk, obwohl in den Handbüchern mancher Hersteller angegeben wird, der Video RAM lasse sich auch auf Kassette abspeichern.
Die Abspeicherung geschieht jedoch aus einem Disk Driver heraus, der erst aktiv wird, sobald die Diskette oder Quick Disk angesprochen und somit auf Kassette nicht wirksam wird.
Wie es scheint gibt es also keine Chance die Video Bilder auf Kassette abzuspeichern.
SCREENSAUE ermöglicht aber jetzt die Abspeicherung von Video Bildern auch auf Kassette, wobei es völlig gleichgültig ist, welcher MSX Computer benutzt wird und welche Peripherie angeschlossen ist, da dies alles vom Programm festgestellt und angepaßt wird.
SCREENSAVE macht sich 2 verschiedene Transportroutinen zu Nutzen, welche im MSX ROM enthalten sind, um Teile des Video RAMs in den Benutzer RAM zu kopieren und umgekehrt.
Tippen Sie das Programm ein, und sichern Sie dieses auf Kassette im ASCII Format (SAVE ”CAS:SCREEN”) Dies hat den Vorteil, daß Sie das Programm jetzt auf jedes bereits bestehende Program mit MERGE aufmischen können.
Wollen Sie jetzt in einem Programm den Bildschirmspeicher wegsichern, so ordnen Sie der Variablen B$ den Namen des zu sichernden Bildes (max. 5 Buchstaben) zu und rufen Sie das Unterprogramm 65000 auf.
Beispiel: 50 B$=”BILD”:
GOSUB 65000
Der komplette Video RAM wird jetzt in 2 Blöcken zu je 8 kBytes gesichert. Auf der Kassette haben die beiden Blöcke nun die Namen BILD1 und BILD2 (wenn wir B$ den Namen BILD zugeordnet hatten).
Um das abgespeicherte Bild jetzt wieder im Programm einzuladen, gehen Sie wie folgt vor:

1. Schalten Sie mit der SCREEN-Anweisung in den Bildschirmmodus, in welchem Sie das Bild abgespeichert haben.
2. Um das Bild zu laden, geben Sie die Programmzeile: BLOAD”CAS:BILD1”,R:- BLOAD”CAS:BILD2”,R ein und das Bild wird wieder von Kassette geladen und angezeigt.

Um ein Bild wieder zu laden, muß SCREENSAVE nicht mit im Rechner geladen sein, da es bei der Bildspeicherung gleich das Maschinenladeprogramm mit wegschreibt.

ANMERKUNG:
Während SCREENSAVE ein Bild auf Kassette schreibt, wird der verfügbare Programmspeicher um 8,5 kBytes verringert. D. h., wenn nur ein Kassettenrecorder angeschlossen ist, bleiben nur noch 20 kBytes Programmspeicher frei.
Durch die Auslagerung der Grafiken extern auf Kassette wird jedoch der Programmspeicher nur noch für das eigentliche Verarbeitungs-Programm gebraucht, so daß die ganze aufwendige Grafikerstellung entfällt.
Disketten - oder Quick Disk Benutzer können sowieso die dann zur Verfügung stehenden Screencopy Routinen benutzen.

VORSICHT:
Nach der Programmzeile 10 von SCREENCOPY darf keine CLEAR Anweisung mehr vorkommen, da der Rechner sonst aussteigt. CLEAR Anweisungen müssen immer vor Zeile 10 stehen (evtl. Zeilennummer ändern).

Das entsprechende Listing finden Sie auf Seite 24

```
1 '*******************************
2 '*  SCREENSAVE auf Kassetten   *
3 '*           M  S  X           *
4 '*-----------------------------*
5 '*  (c)1985 by Volker Becker   *
6 '*     Steinbacher Str.10      *
7 '*      6370 Oberursel 6       *
8 '*******************************
9 '
10 SC=(256*PEEK(&HF673)+PEEK(&HF672)):SB=SC-8*1024-500
65000 DATA 33,0,0,17,0,0,1,0,32,195,15,7,201
65010 DATA 33,0,0,17,0,0,1,0,32,195,68,7,201
65020 POKE &HF672,256*(SB/256-INT(SB/256)):POKE &HF673,INT(SB/256)
65090 RESTORE 65000:FOR I=SB TO SB+12:READ J:POKE I,J:NEXT
65100 GOSUB 65500:DEFUSR0=SB:SA=SB+15:POKE SB+4,256*(SA/256-INT(SA/256)):POKE SB
+5,INT(SA/256):X=USR0(1):FOR I=SB TO SB+12:READ J:POKE I,J:NEXT
65110 SA=SB+15:POKE SB+1,256*(SA/256-INT(SA/256)):POKE SB+2,INT(SA/256)
65120 BSAVE"CAS:"+B$+"1",SB,SB+1024*8+15
65130 GOSUB 65500:DEFUSR=SB:POKE SB+2,32:SA=SB+15:POKE SB+4,256*(SA/256-INT(SA/
256)):POKE SB+5,INT(SA/256):X=USR0(1):FOR I=SB TO SB+12:READ J:POKE I,J:NEXT
65140 SA=SB+15:POKE SB+1,256*(SA/256-INT(SA/256)):POKE SB+2,INT(SA/256):POKE SB+
5,32
65150 BSAVE"CAS:"+B$+"2",SB,SB+1024*8+15
65155 POKE &HF672,256*(SC/256-INT(SC/256)):POKE &HF673,INT(SC/256)
65160 RETURN
65500 RESTORE 65000:FOR I=SB TO SB+12:READ J:POKE I,J:NEXT:RETURN
```

Multicolor-Zeichensatz – schnell und leicht erstellt!

ATARI-XL-Programm: Zeichensteller für GR.12+13

Um in ATARI-Graphics 12 oder 13 vierfarbige Zeichen zu erstellen, bedarf es einer anderen Technik als im normalen Graphics-Bereich, da hier jedes zweite Bit für die Farbinformation verwendet wir. Hier wird nicht auf einer 8 x 8, sondern auf einer 4 x 8 Matrix, ein Zeichen erstellt.
Nach Starten des Programms wird der Zeichensatz in einen anderen Speicherbereich eingelesen. Danach erscheint der Arbeitsscreen: Die 4 x 8 Matrix. Rechts davon die aktuellen Data-Werte, für diejenigen, die sich die neuen Zeichen nur notieren wollen.
Linkes daneben die aktuelle Zeichenfarbe.
Darunter das neue Zeichen in GR.12 bzw. 13 in Originalgröße. Im Textfenster erscheint das Menü:
- umschalten auf Veränderung aller Farben
- automatisches Erstellen der Datas
- listen der 'Zeichen-Datas' auf Cassette oder Diskette. (Können später mit ENTER"C:" bzw. ENTER"D:CHARSET" in jedes Programm eingefügt werden.)
- dateneingabe

Mit dem Joystick kann der Cursor auf jeden beliebigen Punkt der Matrix gebracht werden und durch Drücken des Feuerknopfes kann ein Punkt in der, durch die Tasten 0-4 vorbestimmten Farbe, gesetzt werden.
Nach Umschalten auf den Farbänderungsmodus wirkt der Joystick in links-rechts Richtung als Helligkeitsveränderer und oben-unten verändert das, durch Drücken des Feuerknopfes, vorbestimmte Farbregister.
Im Dateneingabemodus kann man 8 Daten eingeben. Das entsprechende Zeichen erscheint dann auf der Matrix und kann per Joystick weiterverarbeitet werden.
Zu beachten wäre noch, daß die neuen Datas ab Zeilennummer 29200 beginnen!
Ich wünsche allen Programmierern, die von dieser hervorragenden Möglichkeit des ATARI Gebrauch machen wollen, nämlich vierfarbige Zeichen zu erstellen, viel Spaß!!!

Variablentabelle

T,0,01,02,03	
U,N,M,B,F,ZAEHL	
RRR,BBB	ZÄHLVARIABLEN
DAT	FÜR DATENEINGABE
RECH	BINÄRUMRECHNUNG
TEXT	TEXTFENSTER
SCREEN	BILDSCHIRMAUFBAU
CASS	DISK/CASS AUFNAHME
GENE	DATAZEILE GENERIEREN
FARBE	FARBÄNDERUNG
CHAR	NEUER CHARACTERSET
ZEIL	LETZTE DATAZEILE
ZEIL1	ERSTE DATAZEILE
TR	OPTION,START etc.
TAS	LETZTE TASTATUREINGABE
W,E	CURSOR IM TEXTFENSTER
X,Y,JJ,HH	CURSOR AUF MATRIX
RE,FA,HE	FARBVARIABLEN
BIT,BB	FARBWERTE
CH	ADRESSE CHARSET
Z	FARBE UNTER CURSOR
BB	AKTUELLER DATAWERT
CC	AKTUELLE CURSORFARBE
DL	DISPLAYLIST
S	JOYSTICK
ST	FEUERKNOPF
CD$	DISK/CASS
Q$	EINGABE
BIN$	BINÄRZAHL


```
1 REM ***ZEICHEN-ERSTELLER GR.12+13***
2 REM *      ATARI-XL  6.4 KB          *
3 REM *      INGO  PLUECKHAHN          *
4 REM *ALTE RHEINSTR43-41 DUISBURG 1*
5 REM ******************************
29005 T=29000:TEXT=T+77:SCREEN=T+47:CASS=T+105:GENE=T+98:FAR
BE=T+86:DAT=T+60:CHAR=T+122:RECH=T+69:ZEIL=T+201
29006 O=0:O1=1:O2=O1+O1:O3=O2+O1:TR=53279:TAS=0:W=656:E=657:
U=0:ZEIL1=ZEIL
29007 COM RE(O3),FA(O3),HE(O3),BIT(4,8),BB(O1),Q$(O1),CD$(8)
,BIN$(8):POKE 731,O1
29008 FOR N=O TO O2:RE(N)=N:FA(N)=N+O2:HE(N)=N*5+O2:NEXT N
29009 GOSUB CHAR:GOSUB SCREEN:GOSUB TEXT:GOSUB 29015
29010 FOR M=O1 TO 4:FOR N=O1 TO 8:BIT(M,N)=0:NEXT N:NEXT M
29011 FOR B=O TO 7:POKE CH+B+72,0:NEXT B
29012 GOTO 29032
29013 FOR N=O TO 30:NEXT N:SOUND 0,0,0,0:RETURN
29014 POKE W,O2:POKE E,33:? CHR$(TAS+176)
29015 COLOR 40*(TAS=0)+8*(TAS=01)+168*(TAS=02)+159*(TAS>4 OR
 TAS<O OR TAS=03):PLOT O2,5:RETURN
29016 POKE 842,12:GOSUB TEXT:POKE 559,34:GOTO 29034
29017 REM *******TASTE GEDRUECKT ?****
29018 POKE 77,0:POKE 694,0
29019 CLOSE #01:OPEN #01,4,0,"K:":GET #01,TAS:TAS=TAS-48
29020 IF ((TAS<O OR TAS>4) OR TAS=03) THEN SOUND 0,10,12,15:
GOSUB 29015:GOSUB 29013:GOTO 29018
29021 GOSUB 29014:RETURN
29022 REM *******ZEICHEN POKEN********
29023 LOCATE X,Y,Z
29024 IF X=5 THEN BIT(01,Y)=192*(Z=168)+128*(Z=8)+64*(Z=40)
29025 IF X=6 THEN BIT(02,Y)=48*(Z=168)+32*(Z=8)+16*(Z=40)
29026 IF X=7 THEN BIT(03,Y)=12*(Z=168)+8*(Z=8)+4*(Z=40)
29027 IF X=8 THEN BIT(4,Y)=03*(Z=168)+02*(Z=8)+(Z=40)
29028 BB=BIT(01,Y)+BIT(02,Y)+BIT(03,Y)+BIT(4,Y):POSITION 12,
Y:? #6;BB;"  ";
29029 FOR F=12 TO 14:LOCATE F,Y,Z:Z=Z+96:COLOR Z:PLOT F,Y:NE
XT F:IF ZAEHL THEN RETURN
29030 FOR N=O1 TO 8:IF Y=N THEN POKE CH+71+N,BB:NEXT N:COLOR
 41:PLOT 19,13:PLOT 19,18:RETURN
29031 NEXT N:RETURN
29032 X=8:Y=01:GOSUB 29014
29033 REM *******HAUPTSCHLEIFE********
29034 POKE 77,0:IF PEEK(764)<>255 THEN GOSUB 29017
29035 S=STICK(O):CC=40*(TAS=O)+8*(TAS=01)+168*(TAS=02)+32*(T
AS=4)
29036 X=X+(S=7)*(X<8)-(S=11)*(X>5)
29037 Y=Y+(S=13)*(Y<8)-(S=14)*(Y>01)
29038 IF  NOT STRIG(O) THEN COLOR CC:PLOT X,Y:GOSUB 29022:PO
P :SOUND 0,200,10,10:GOSUB 29013:GOTO 29034
29039 LOCATE X,Y,Z:COLOR 139:PLOT X,Y:COLOR Z:PLOT X,Y
```

```
29040 IF PEEK(TR)=5 THEN GOTO GENE
29041 IF PEEK(TR)=6 THEN GOSUB FARBE:GOSUB TEXT:GOSUB 29014
29042 IF PEEK(TR)=03 THEN GOSUB CASS:GOSUB TEXT
29043 IF PEEK(732)=17 THEN POKE 732,0:GOSUB DAT:GOSUB TEXT:G
OTO 29039
29044 COLOR 139:PLOT X,Y:COLOR Z:PLOT X,Y
29045 GOTO 29034
29046 REM *********BILDSCHIRM**********
29047 GRAPHICS 01:POKE 752,01:POKE 756,CH/256:POKE 82,0:FOR
N=0 TO 02:SETCOLOR RE(N),FA(N),HE(N)
29048 NEXT N:SETCOLOR 03,0,12
29049 POKE 16,64:POKE 53774,64
29050 DL=PEEK(560)+256*PEEK(561):POKE DL+18,4:POKE DL+22,5:P
OKE DL+24,02
29051 COLOR 144:FOR N=01 TO 8:PLOT 12,N:NEXT N
29052 COLOR 141:PLOT 4,0:DRAWTO 4,9:DRAWTO 9,9:DRAWTO 9,0:DR
AWTO 4,0
29053 FOR N=11 TO 16 STEP 5:POSITION 4,N:? #6;"graphics":NEX
T N
29054 COLOR 145:PLOT 13,11:PLOT 13,16
29055 COLOR 146:PLOT 14,11:COLOR 147:PLOT 14,16
29056 COLOR 41:PLOT 19,13:PLOT 19,18
29057 POSITION 0,21:? #6;" |          MENUE
     |"
29058 RETURN
29059 REM ****DATEN EINGEBEN**********
29060 HH=X:JJ=Y:? "K":POKE W,0:POKE E,12:? "GIB 8 DATEN EIN"
:POKE W,03:POKE E,5:? "ZURUECK ZUM PROGRAMM--RETURN";
29061 FOR Y=01 TO 8:POKE W,01:POKE E,17:TRAP 29065:POKE W,01
:POKE E,16:? Y
29062 POKE W,01:POKE E,17:INPUT BB:IF BB>255 THEN POKE W,01:
POKE E,18:? "   ":GOTO 29062
29063 POKE W,01:POKE E,18:? "   ":POSITION 12,Y:? #6;BB;"  "
29064 GOSUB RECH:GOSUB 29029:NEXT Y
29065 ? "K↓↓      EINEN AUGENBLICK BITTE"
29066 ZAEHL=01:FOR Y=01 TO 8:FOR X=5 TO 8:GOSUB 29023:NEXT X
:NEXT Y:ZAEHL=0
29067 GOSUB 29013:X=HH:Y=JJ:RETURN
29068 REM ******BINAER-UMRECHNUNG******
29069 RRR=BB
29070 F=128:FOR N=01 TO 8
29071 BIN$(N,N)=CHR$(48+((BB-F)>=0)):BB=BB-F*(BB-F)=0):F=F/0
2
29072 NEXT N:BB=RRR
29073 FOR F=0 TO 6 STEP 02
29074 CC=40*BIN$(F+01,F+02)="01"+8*BIN$(F+01,F+02)="10"+168*
BIN$(F+01,F+02)="11"+32*BIN$(F+01,F+02)="00"
29075 COLOR CC:PLOT F/02+5,Y:NEXT F:RETURN
29076 REM ******TEXTFENSTER**********
29077 ? "K":POKE W,0:POKE E,0
29078 ? " |  START-FARBEN AENDERN"
29079 ? " | SELECT-DATAS ERSTELLEN"
29080 ? " | OPTION-LIST AUF CASS/DISK"
29081 ? " |   HELP-DATEN EINGEBEN";
29082 FOR N=0 TO 03:POKE W,N:POKE E,29:? "|          |";:NEXT
N:POKE W,0:POKE E,30:? "REGISTER"
29083 POKE W,01:POKE E,30:? "  [0-4]  "
29084 TAS=0:GOSUB 29014:RETURN
29085 REM ******SETCOLOR AENDERN********
29086 ? "K |STICK LI-RE-HELLIGKEIT":? " |STICK OB-UN-FARBWER
T"
```



```
29087 ? " |   KNOPF   -REGISTER":? " |   START   -ZUM PROGRA
MM";
29088 GOSUB 29082:POKE W,01:POKE E,30:? "SETCOLOR":U=0
29089 S=STICK(0):ST=STRIG(0)
29090 FA(U)=FA(U)+(S=13)*(FA(U)<15)-(S=14)*(FA(U)>0)
29091 HE(U)=HE(U)+(S=7)*(HE(U)<15)-(S=11)*(HE(U)>0)
29092 IF  NOT ST THEN SOUND 0,100,10,10:GOSUB 29013:U=U+01:I
F U>02 THEN U=0
29093 POKE W,02:POKE E,30:? RE(U):POKE W,02:POKE E,32:? FA(U
);"■":POKE W,02:POKE E,35:? HE(U);"■"
29094 SETCOLOR RE(U),FA(U),HE(U)
29095 IF PEEK(TR)=6 THEN RETURN
29096 POKE 77,0:GOTO 29089
29097 REM ***DATAZEILE GENERIEREN*****
29098 POKE 559,0:? "K":? :? ZEIL;"D.";:FOR M=01 TO 8
29099 BBB=BIT(01,M)+BIT(02,M)+BIT(03,M)+BIT(4,M)
29100 ? BBB;:IF M=8 THEN ? :GOTO 29102
29101 ? ",";:NEXT M
29102 ZEIL=ZEIL+01:POKE W,03:POKE E,0:? "G.29016";
29103 POKE W,0:POKE E,0:POKE 842,13:STOP
29104 REM ****AUF CASS/DISK LISTEN****
29105 POKE 752,01:TRAP 29119:? "K":? :? " AUF CASSETTE ODER
DISKETTE -C/D-";:INPUT Q$
29106 IF Q$="C" THEN CD$="CASSETTE"
29107 IF Q$="D" THEN CD$="DISKETTE"
29108 IF Q$<>"C" AND Q$<>"D" THEN 29105
29109 ? "K":? "↓   LEGE EINE ";CD$;" EIN":? "↓  WENN FERTIG
DRUECK START":?
29110 ? "   ZURUECK ZUM PROGRAMM DRUECK OPTION";
29111 IF PEEK(TR)=6 THEN 29114
29112 IF PEEK(TR)=03 THEN RETURN
29113 GOTO 29111
29114 ? "K":? ,"ACHTUNG AUFNAHME":? ,"  BITTE WARTEN";
29115 IF Q$="C" THEN 29117
29116 LIST "D:CHARSET.BAS",ZEIL1,ZEIL:RETURN
29117 POKE 764,12:LIST "C",ZEIL1,ZEIL
29118 SOUND 0,0,0,0:RETURN
29119 ? "K":? :? ,"DA STIMMT WAS NICHT !":FOR N=0 TO 200:SET
COLOR 4,N,N:NEXT N
29120 SETCOLOR 4,0,0:GOTO 29105
29121 REM *********CHARSET**********
29122 CH=(PEEK(106)-8)*256:POKE 559,0
29123 U=USR(ADR("hh▪Lh▪Kh▪Nh▪M" ■♥JK▪M▪PYFLFNJPP♦"),57344,CH
)
29124 FOR B=0 TO 7:POKE CH+B+64,255:POKE CH+B+72,0:NEXT B
29125 POKE 559,34:RETURN
29200 REM ↓ DATAS BEGINNEN HIER ↓
```


IMPRESSUM

Tips & Tricks von CPU
erscheint alle 2 Monate im
Tronic-Verlag, 3444 Wehretal 1
Tel. 05651/4 06 93 oder 4 06 43

Herausgeber:
Axel Credé
Chefredakteur:
Siegfried Görk
Redaktion:
Manfred Kleimann, Hartmut Wendt, Frank Brall, Ottfried Schmidt.

Programmautoren:
V. Becker, D. Durstewitz, A. Fuhrig, R. Gawrikow, M. de Hogh, E. Hölscher, W. Huslik, M. Kotulla, W. Krieg, G. Michel, U. Putz, W. Raaz, K. Rupprecht, P. Schmitz, W. Sohrt, O. Steinmeyer, H. Thöne, D. Twigg-Flesner, H. Wenz u. a.

Gesamtherstellung:
Druckhaus Dierichs Kassel, Frankfurter Str. 168, 3500 Kassel

Vertrieb:
Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz:
Verlagsunion
Friedrich-Bergius-Straße 20
6200 Wiesbaden
Telefon 06121/2660

Anzeigenverwaltung:
Markt-Kommunikation
GmbH + Co. KG
Hohenzollernstr. 60, 8000 München 40,
Telefon 089/33 40 13 14, Telex 05 29 212
BTX Leitseite * 31880 /

Anzeigenleitung:
Susanne Ehlers
Hohenzollernstraße 60
8000 München 40

Erscheinungsweise:
Erstverkaufstag von
Tips & Tricks von CPU ist Anfang des Monats

Anzeigenpreisliste:
Bitte Mediaunterlagen anfordern.

Software-Service und Vertrieb:
Annelie Kratzenberg, Heike Rabe

Urheberrecht:
Alle in HOMECOMPUTER veröffentlichten Beiträge sind urheberrechtlich geschützt. Alle Rechte, auch Übersetzungen, vorbehalten.
Reproduktionen jeder Art (Fotokopien, Microfilm, Erfassung in Datenverarbeitungsanlagen usw.) bedürfen der schriftlichen Genehmigung des Verlags. Alle veröffentlichte Software wurde von Mitarbeitern des Verlages oder von freien Mitarbeitern erstellt.
Aus Ihrer Veröffentlichung kann nicht geschlossen werden, daß die beschriebenen Lösungen oder Bezeichnungen frei von Schutzrechten sind.

Bezugspreise:
Einzelheft 6,— DM,
Abonnements Ausland:
Tips & Tricks von CPU (6 Hefte):
Europäisches Ausland 43,-- DM
Außereuropäisches Ausland 60,-- DM
Inlands-Abonnement
von Tips & Tricks von CPU 30,- DM
Europäisches Ausland 87,50 DM
Außereuropäisches Ausland 120,-- DM
Super-Kombi:
HOMECOMPUTER (6 Hefte) &
COMPUTRONIC (6 Hefte) & TIPS & TRICKS von CPU:
Europäisches Ausland 127,50 DM
Außereuropäisches Ausland 170,-- DM

Autoren, Manuskripte:
Der Verlag nimmt Manuskripte und Software zur Veröffentlichung gerne entgegen. Sollte keine andere Vereinbarung getroffen sein, so gehen wir davon aus, daß Sie mit einem Honorar von 120,-DM pro abgedruckter Seite im Heft einverstanden sind.

Titelblatt:
Werbeatelier H. Kästle, Eschwege

Nützliche BASIC-Erweiterung für den TI-99

Das Programm STRINGKIT erweitert das Extended Basic des TI-99 um einige nützliche Stringfunktionen, die der Programmierer bisher schmerzlich vermißte. Das Programm ist in Maschinensprache geschrieben und damit auf jeden Fall schneller als Basic-Unterprogramme, die diese Befehle simulieren. STRINGKIT läuft auf dem TI-99/4A mit Extended Basic und 32K-Speichererweiterung.

Bei der Programmeingabe gibt es wie üblich drei Wege: Eingabe des Quellcodes und Assemblieren mit dem Editor/Assembler-Modul, Speichern des Objektcodes in einer Datei des Formats DISPLAY/FIXED 80 oder Eingabe des Basic-Loaders mit den DATA-Zeilen. Allen Besitzern des Editor/Assemblers sei einer der ersten beiden Wege empfohlen, da das Programm als Objektcode im Speicher frei verschiebbar (relokatibel) ist, während die Version mit DATA-Zeilen an den Speicherbereich 9460 bis 10495 gebunden ist.

Entsprechend der Programmspeicherung erfolgt das Einlesen mit CALL LOAD("Datei") oder RUN "Programm". Steht das Programm im Speicher, so wird das Basic um sechs Befehle erweitert, die alle mit CALL LINK aufgerufen werden:

CALL LINK("UPPER$",IN$,OUT$) wandelt im String IN$ alle Kleinbuchstaben in Großbuchstaben um und speichert den neuen String in OUT$. Alle anderen Zeichen im String bleiben unverändert. Der Befehl entspricht der UPPER$-Funktion im CPC 464-Basic und ist nützlich bei Benutzereingaben, die in ein bestimmtes Format gebracht werden sollen.

Beispiel: CALL LINK("UPPER$","ABCabc12",B$):: PRINT B$ liefert "ABCABC12"

CALL LINK("LOWER$",IN$,OUT$) wandelt analog zum UPPER$-Befehl alle Großbuchstaben des Strings IN$ in Kleinbuchstaben um und speichert das Ergebnis in OUT$. Dieser Befehl entspricht der LOWER$-Funktion im CPC 464-Basic.

Beispiel: CALL LINK("LOWER$","ABCabc1;",B$):: PRINT B$ liefert "abcabc12"

CALL LINK("LEFT$",IN$,COUNT,OUT$) entspricht der LEFT$-Funktion im Microsoft-Basic und vielen anderen Basic-Dialekten. Der Befehl nimmt aus dem String IN$ alle Zeichen vom ersten bis zum durch COUNT angegebenen Zeichen und speichert sie in OUT$.

Beispiel: CALL LINK("LEFT$","ABCDEFG",3,B$):: PRINT B$ liefert "ABC"

CALL LINK("RIGHT$",IN$,COUNT,OUT$) entspricht der RIGHT$-Funktion im Microsoft-Basic und nimmt im Gegensatz zu LEFT$ die angegebene Anzahl von Zeichen rechts vom String IN$ ab.

Beispiel: CALL LINK("RIGHT$",ABCDEFG",4,B$):: PRINT B$ liefert "DEFG"

CALL LINK("SWAP",A$(),B$(), COUNT) tauscht die Elemente zweier Stringdatenfelder gegeneinander aus. Dabei werden alle Elemente vom 0. bis 1. Element (entsprechend OPTION BASE) bis zu dem Wert von COUNT entsprechenden Element berücksichtigt. Das Programm beachtet also, ob OPTION BASE 0 oder OPTION BASE 1 gewählt wurde. Bitte beachten Sie, daß den Arrays-Bezeichnungen leere Klammern folgen müssen, Zahlenwerte oder Variablennamen in den Klammern führen zu einer Fehlermeldung.

CALL LINK("SWAP",A(),B(),COUNT) ist das zweite Format des SWAP-Befehls, denn hiermit können auch die Inhalte von Zahlenfeldern ausgetauscht werden. Auch hier gilt, daß der für OPTION BASE gewählte Wert berücksichtigt wird.

Für Maschinenprache-Programmierer sei noch erwähnt, wie STRINGKIT auf den Wert von OPTION BASE kommt. Texas Instruments hat es nirgendwo erwähnt, aber per Zufall entdeckte der Autor in Disassemblieren die Speicherstelle >8343. Sie enthält bei OPTION BASE 0 den Wert 0, bei OPTION BASE 1 entsprechend die Zahl 1. Jeder andere Wert der hier gePOKEt wird, entspricht ebenfalls OPTION BASE 1.

Martin Kotulla

```
100 !  ********************
110 ! **********************
120 ! *                    *
130 ! *    STRINGKIT       *
140 ! *                    *
150 ! **********************
160 !  ********************
170 !
180 ! ------------------------
190 ! (C) Martin Kotulla 85
200 ! ------------------------
210 !
```



```
220 CALL CHAR(42,"AA55AA55AA55AA55",91,"7E8199A1A199817E")
230 CALL CHAR(45,RPT$("0",8)&"FF")
240 CALL CHAR(128,"DFBF60C0919B959191808080C060BFDF")
250 CALL CHAR(130,"FBFD06030101014951615149030 6FDFB")
260 CALL CLEAR :: CALL HCHAR(2,4,42,26)
270 CALL VCHAR(3,4,42,5):: CALL VCHAR(3,29,42,5)
280 CALL MAGNIFY(3):: CALL SPRITE(#1,128,2,28,70)
290 DISPLAY AT(5,12)SIZE(9):"STRINGKIT"
300 CALL HCHAR(8,4,42,26)
310 DISPLAY AT(12,4):CHR$(91)&"  MARTIN KOTULLA 1985"
320 CALL HCHAR(13,6,45,22):: DISPLAY AT(17,7):"DATAs PRUEFEN? J"
330 ACCEPT AT(17,22)BEEP SIZE(-1)VALIDATE("JN"):JN$
340 IF JN$="N" THEN 390
350 FOR I=1 TO 1080 STEP 2 :: READ A1,A2 :: B=B+A1+A2 :: NEXT I
360 DISPLAY AT(20,5):"DIE DATA-ZEILEN SIND"
370 IF B()90212 THEN DISPLAY AT(22,10):"FEHLERHAFT!" :: END
380 DISPLAY AT(22,11):"KORREKT !"
390 CALL PEEK(8198,A):: IF A()170 THEN CALL INIT :: RESTORE ELSE RESTORE
400 FOR I=9460 TO 10495 STEP 5 :: READ A,B,C,D,E
410 CALL LOAD(I,A,B,C,D,E):: NEXT I
420 CALL PEEK(8196,A,B):: C=A*256+B-40
430 CALL LOAD(8196,INT(C/256),C AND 255)
440 FOR I=C TO C+39 :: READ A :: CALL LOAD(I,A):: NEXT I
450 DISPLAY ERASE ALL BEEP :: END
460 !
470 ! MASCHINENCODE:
480 !
490 DATA 255,000,000,000,000,000,000,000,000,000,000,000,203,020,203,053
500 DATA 203,078,203,231,204,071,204,150,204,228,205,029,205,075,205,096
510 DATA 033,131,035,253,038,184,040,183,041,182,042,195,043,193,044,179
520 DATA 045,194,047,196,058,181,059,180,060,191,061,190,062,192,094,197
530 DATA 255,058,058,130,065,084,240,071,079,133,073,070,132,079,078,155
540 DATA 079,082,186,080,073,221,084,079,177,255,065,066,083,203,065,076
550 DATA 076,236,065,078,068,187,065,083,067,220,065,084,078,204,066,089
560 DATA 069,003,067,079,078,001,067,079,083,205,068,069,070,137,068,073
570 DATA 077,138,069,078,068,139,069,079,070,202,069,088,080,206,070,079
580 DATA 082,140,073,078,084,207,076,069,078,213,076,069,084,141,076,079
590 DATA 071,208,077,065,088,223,077,073,078,224,078,069,087,000,078,079
600 DATA 084,189,078,085,077,004,079,076,068,005,080,079,083,217,082,069
610 DATA 067,222,082,069,077,154,082,069,083,006,082,078,068,215,082,085
620 DATA 078,169,083,071,078,209,083,073,078,210,083,081,082,211,083,085
630 DATA 066,161,084,065,066,252,084,065,078,212,086,065,076,218,088,079
640 DATA 082,188,255,066,065,083,069,241,066,069,069,080,238,067,065,076
650 DATA 000,255,067,072,082,036,214,068,065,084,065,147,038,066,200,001
660 DATA 037,178,200,002,037,180,004,096,048,094,004,224,038,066,004,096
670 DATA 048,124,004,192,002,008,224,129,216,008,131,212,002,008,160,129
680 DATA 216,008,140,002,006,200,216,008,140,002,004,224,038,066,004,224
690 DATA 034,160,192,000,019,002,004,096,034,018,004,096,034,032,000,000
700 DATA 007,096,047,164,022,007,007,032,034,152,200,001,038,086,004,032
710 DATA 043,074,000,000,004,091,007,032,034,156,006,160,036,058,006,160
720 DATA 038,106,004,096,035,016,007,096,047,164,022,244,007,032,034,152
730 DATA 002,009,041,172,002,008,032,032,002,006,000,080,222,072,006,006
740 DATA 021,253,200,001,038,142,004,032,043,116,000,000,041,172,192,000
750 DATA 019,225,004,096,037,184,200,004,034,158,006,160,036,052,193,032
760 DATA 034,158,004,224,034,158,160,135,016,220,007,032,034,156,006,160
```

```
 770 DATA 036,058,006,001,128,193,017,006,002,034,255,170,004,224,034,152
 780 DATA 004,096,035,016,192,195,022,001,004,193,192,193,016,249,000,000
 790 DATA 000,007,000,012,000,025,000,030,000,045,000,059,000,079,255,255
 800 DATA 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255
 810 DATA 255,255,255,000,000,006,002,011,035,016,194,002,098,032,034,142
 820 DATA 194,072,098,004,017,006,200,008,038,248,002,136,000,040,020,001
 830 DATA 004,091,004,196,002,224,038,248,006,160,040,152,002,001,000,002
 840 DATA 006,160,040,138,192,096,131,074,209,032,036,244,009,132,129,001
 850 DATA 018,002,004,096,040,198,006,193,216,001,036,244,002,002,036,244
 860 DATA 004,096,040,212,002,224,038,248,006,160,040,152,002,001,000,002
 870 DATA 006,160,040,138,192,096,131,074,208,160,036,244,009,130,128,066
 880 DATA 020,002,004,096,040,198,096,129,002,004,036,244,160,132,006,193
 890 DATA 212,129,004,096,040,212,002,224,038,248,006,160,040,152,002,000
 900 DATA 036,245,208,096,036,244,009,129,208,144,009,130,002,130,000,097
 910 DATA 017,005,002,130,000,122,021,002,002,034,255,224,006,194,220,002
 920 DATA 006,001,024,242,004,192,002,001,000,002,002,002,036,244,004,032
 930 DATA 032,016,004,096,040,222,002,224,038,248,006,160,040,152,002,000
 940 DATA 036,245,208,096,036,244,009,129,208,144,009,130,002,130,000,065
 950 DATA 017,005,002,130,000,090,021,002,002,034,000,032,006,194,220,002
 960 DATA 006,001,024,242,004,192,002,001,000,002,002,002,036,244,004,032
 970 DATA 032,016,004,096,040,222,002,224,038,248,002,001,000,003,006,160
 980 DATA 040,138,193,160,131,074,210,032,131,000,009,136,002,136,000,005
 990 DATA 019,025,208,032,131,067,009,128,006,065,004,032,032,012,006,160
1000 DATA 040,168,005,129,004,032,032,012,006,001,004,032,032,008,006,160
1010 DATA 040,178,005,129,004,032,032,008,005,128,006,001,006,006,024,237
1020 DATA 004,096,040,222,192,198,208,032,131,067,009,128,002,001,000,001
1030 DATA 002,002,036,244,212,160,038,246,004,032,032,020,005,129,002,002
1040 DATA 037,245,212,160,038,246,004,032,032,020,006,001,004,032,032,016
1050 DATA 005,129,002,002,036,244,004,032,032,016,005,128,006,001,006,003
1060 DATA 024,231,004,096,040,222,004,192,004,032,032,012,004,032,032,024
1070 DATA 018,184,004,091,004,192,002,001,000,001,002,002,036,244,004,032
1080 DATA 032,020,004,091,002,003,131,074,002,004,036,245,016,004,002,003
1090 DATA 036,245,002,004,131,074,002,005,000,004,205,051,006,005,022,253
1100 DATA 004,091,216,032,038,246,036,244,002,000,030,000,004,032,032,052
1110 DATA 004,192,002,001,000,003,004,032,032,016,216,032,038,246,036,244
1120 DATA 216,032,038,246,037,245,216,032,038,247,131,124,002,224,131,224
1130 DATA 004,096,000,112,040,020,200,012,041,014,016,002,200,008,041,014
1140 !
1150 ! NAMENSTABELLE:
1160 !
1170 DATA 083,087,065,080,032,032,039,250,076,079,087,069,082,036,039,186
1180 DATA 085,080,080,069,082,036,039,122,082,073,071,072,084,036,039,072
1190 DATA 076,069,070,084,036,032,039,024
1200 END ! ************************************************************
```

```
*   ********************************
*  **********************************
* *                                *
* *          STRINGKIT             *
* *                                *
*  **********************************
*   ********************************
*
*
*   (C)  Martin Kotulla - 7.5.1985
*   ------------------------------
```



```
* Neue Befehle:
*
* - CALL LINK("UPPER$",in$,out$)
* - CALL LINK("LOWER$",in$,out$)
*
* - CALL LINK("LEFT$",in$,cnt,out$)
* - CALL LINK("RIGHT$",in$,cnt,out$)
*
* - CALL LINK("SWAP",io1(),io2(),cnt)
* - CALL LINK("SWAP",io1$(),io1$(),cnt)
*
* in$         = Eingabestring
* out$        = Ausgabestring
* cnt         = Zähler
* io1/io2     = Eingabe und Ausgabe
* io1$/io2$   = Ein- und Ausgabestring
*
```

Achtung! - Achtung!

In dieser Ausgabe Super Software-Angebote!

Mehr darüber auf Seite 2!

```
       DEF  LEFT$,RIGHT$
       DEF  UPPER$,LOWER$
       DEF  SWAP

       IDT  'STRKIT'

NEXT   EQU  >0070

ARGIDT EQU  >8300          Argument-Identifier bei CALL LINK-Aufruf
OPBASE EQU  >8343          Wert von Option Base
FAC    EQU  >834A          Fließkomma-Akkumulator
STATUS EQU  >837C          GPL-Statusbyte
GPLWS  EQU  >83E0          GPL-Workspace

NUMASG EQU  >2008          Numerischen Parameter an Basic übergeben
NUMREF EQU  >200C          Numerischen Parameter aus Basic holen
STRASG EQU  >2010          String an Basic übergeben
STRREF EQU  >2014          String aus Basic holen
XMLLNK EQU  >2018          ROM-Routinen aufrufen
ERR    EQU  >2034          Fehlermeldung ausgeben

ERRBV  EQU  >1E00          Code für BAD VALUE
CFI    EQU  >12B8          Convert Floating Point To Integer

BUF1   BYTE 255
BUF2   BSS  256
BUF3   BYTE 255
BUF4   BSS  256

HEXFF  BYTE >FF
HEX00  BYTE >00

USRWS  BSS  32             Workspace des Programms

*********************************************************************************

LEFT$  LWPI USRWS          Eigene Workspace-Register laden
       BL   $GETSTR        String aus CALL LINK holen
```

```
LFTLEN LI   R1,2          Längenangabe aus CALL LINK holen
       BL   $GETNUM
       MOV  $FAC,R1

LFTCHK MOVB $BUF1,R4      Stringlänge des übergebenen Strings holen
       SRL  R4,8          ... und ins LSByte schieben
       C    R1,R4         Stringlänge kleiner als CALL LINK-Angabe?
       JLE  LFTCUT
       B    $BADVAL       Ja - Bad Value ausgeben

LFTCUT SWPB R1            Dem Längenbyte neuen Wert zuweisen
       MOVB R1,$BUF1
       LI   R2,BUF1       Den String an Basic übergeben
       B    $ASGSTR       Rücksprung nach Basic

*****************************************************************************

RIGHT$ LWPI USRWS         Eigene Workspace-Register laden
       BL   $GETSTR       String aus CALL LINK holen

RGTLEN LI   R1,2          Längenwert aus CALL LINK holen
       BL   $GETNUM
       MOV  $FAC,R1

RGTCHK MOVB $BUF1,R2      Tatsächliche Länge des Strings holen
       SRL  R2,8
       C    R2,R1         Mit Wert aus CALL LINK vergleichen
       JHE  RTCALC
       B    $BADVAL       Argument zu klein: Bad Value ausgeben

RTCALC S    R1,R2         Stringlänge:=Stringlänge-Argumentwert
       LI   R4,BUF1
       A    R4,R2         Adresse:=Bufferstart+Stringlänge

RGTCUT SWPB R1            Neue Stringlänge:=Argumentwert
       MOVB R1,*R2        *R2 wird neuer Bufferbeginn
       B    $ASGSTR       String zuweisen, Basic-Rücksprung

*****************************************************************************

UPPER$ LWPI USRWS         Eigene Workspace-Register laden
       BL   $GETSTR       String aus CALL LINK holen

UPRLEN LI   R0,BUF2       Zeiger auf den String-Buffer
       MOVB $BUF1,R1      Längenbyte aus dem Buffer holen
       SRL  R1,8          Und ins LSByte schieben

USHTLP MOVB *R0,R2        Einen Buchstaben aus dem Stringbuffer holen
       SRL  R2,8          Und ins LSByte schieben
       CI   R2,'a'        Kleiner als "a"?
       JLT  UNOSHT        Ja - nicht shiften
       CI   R2,'z'        Grösser als "z"?
       JGT  UNOSHT        Ja - nicht shiften

USHIFT AI   R2,-32        ASCII-Code -32 ergibt Großbuchstaben
```



```
UNOSHT SWPB R2            Buchstaben wieder ins MSByte zurück
       MOVB R2,*R0+       Und wieder in den Buffer schreiben

USNEXT DEC  R1            Längenzähler dekrementieren
       JOC  USHTLP        Fertig? Nein - weitermachen

UPASG$ CLR  R0            String wieder an Basic zuweisen: einfache Variable
       LI   R1,2          - Zweiter Parameter in CALL LINK
       LI   R2,BUF1       - Zeiger auf Stringbuffer
       BLWP $STRASG       - String zuweisen
       B    $RETURN       Rücksprung nach Basic

*****************************************************************************

LOWER$ LWPI USRWS         Eigene Workspace-Register laden
       BL   $GETSTR       String aus CALL LINK holen

LWRLEN LI   R0,BUF2       Zeiger auf den String-Buffer
       MOVB $BUF1,R1      Längenbyte aus dem Buffer holen
       SRL  R1,8          Und ins LSByte schieben

LSHTLP MOVB *R0,R2        Einen Buchstaben aus dem Stringbuffer holen
       SRL  R2,8          Und ins LSByte schieben
       CI   R2,'A'        Kleiner als "A"?
       JLT  LNOSHT        Ja - nicht shiften
       CI   R2,'Z'        Grösser als "Z"?
       JGT  LNOSHT        Ja - nicht shiften

LSHIFT AI   R2,32         ASCII-Code +32 ergibt Kleinbuchstaben

LNOSHT SWPB R2            Buchstaben wieder ins MSByte zurück
       MOVB R2,*R0+       Und wieder in den Buffer schreiben

LSNEXT DEC  R1            Längenzähler dekrementieren
       JOC  LSHTLP        Fertig? Nein - weitermachen

LWASG$ CLR  R0            String wieder an Basic zuweisen: einfache Variable
       LI   R1,2          - Zweiter Parameter in CALL LINK
       LI   R2,BUF1       - Zeiger auf Stringbuffer
       BLWP $STRASG       - String zuweisen
       B    $RETURN       Rücksprung nach Basic

*****************************************************************************

SWAP   LWPI USRWS         Eigene Workspace-Register laden
       LI   R1,3          Zahl der Array-Elemente holen
       BL   $GETNUM
       MOV  $FAC,R6

STRNUM MOVB $ARGIDT,R8    Argument Identifier holen
       SRL  R8,8          Und ins LSByte schieben
       CI   R8,5          Ist es ein String-Array?
       JEQ  SWAP$         Ja - nach SWAP$ springen

SWPNUM MOVB $OPBASE,R0    Option Base 0 oder 1
       SRL  R0,8          Ergebnis ins LSByte schieben
       DECT R1            Zeiger auf erstes Argument in CALL LINK
```

```
SWNULP BLWP $NUMREF        Zahl eines Feldelements holen
       BL   $FACBUF        Im internen Buffer zwischenspeichern
       INC  R1             Zeiger auf zweiten Array
       BLWP $NUMREF        Zahl eines Feldelements aus 2.Array holen
       DEC  R1             Zeiger wieder auf 1.Array
       BLWP $NUMASG        Und Zahl dem 1.Array zuweisen
       BL   $BUFFAC        Vorherige Zahl aus dem Buffer in den FAC holen
       INC  R1             Zeiger wieder auf 2.Array
       BLWP $NUMASG        Und Zahl einem Element des 2.Arrays zuweisen

NXTNUM INC  R0             Die nächsten beiden Elemente der Felder
       DEC  R1             Wieder Zeiger auf erstes Argument
       DEC  R6             Zähler dekrementieren
       JOC  SWNULP         Fertig? Nein - weitermachen
       B    $RETURN        Rücksprung nach Basic

*******************************************************************************

SWAP$  MOV  R6,R3          Zahl der Array-Elemente kopieren
       MOVB $OPBASE,R0     Option Base 0 oder 1
       SRL  R0,8           Ergebnis ins LSByte schieben
       LI   R1,1           Zeiger auf erstes Argument in CALL LINK

GET1$  LI   R2,BUF1        String aus dem 1.Array in CALL LINK holen
       MOVB $HEXFF,*R2
       BLWP $STRREF

GET2$  INC  R1             String aus dem 2.Array in CALL LINK holen
       LI   R2,BUF3
       MOVB $HEXFF,*R2
       BLWP $STRREF

GIVE1$ DEC  R1             Zeiger auf 1.Array
       BLWP $STRASG        Dem 1.Array den String des 2.Arrays zuweisen

GIVE2$ INC  R1             Zeiger auf 2.Array
       LI   R2,BUF1        Zeiger auf String des 1.Arrays
       BLWP $STRASG        Dem 2.Array den String des 1.Arrays zuweisen

NXTSTR INC  R0             Zeiger auf nächste Array-Elemente
       DEC  R1             Zeiger auf 1.Array in CALL LINK
       DEC  R3             Schleifenzähler dekrementieren
       JOC  GET1$          Fertig? Nein - weitermachen
       B    $RETURN        Rücksprung nach Basic

*******************************************************************************

GETNUM CLR  R0             Zahl aus CALL LINK holen
       BLWP $NUMREF
       BLWP $XMLLNK        Und in Integer umwandeln
       DATA CFI
       RT

GETSTR CLR  R0             String aus CALL LINK holen
       LI   R1,1
       LI   R2,BUF1
```



```
       BLWP $STRREF
       RT

*******************************************************************************

FACBUF LI   R3,FAC         Fließkomma-Zahl von FAC in einen Buffer kopieren
       LI   R4,BUF2
       JMP  FTRANS

BUFFAC LI   R3,BUF2        Fließkomma-Zahl von einem Buffer in FAC kopieren
       LI   R4,FAC
FTRANS LI   R5,4           8 Bytes = 4*16-Bit Words

FTLOOP MOV  *R3+,*R4+      Kopierschleife
       DEC  R5             Schleifenzähler herunterzählen
       JNE  FTLOOP         Fertig? Nein - weitermachen
       RT                  Rücksprung zum aufrufenden Programm

*************************************

BADVAL MOVB $HEXFF,$BUF1   Maximale Stringlänge in den Buffer schreiben
       LI   R0,ERRBV       Errorcode von BAD VALUE laden
       BLWP $ERR           Fehlermeldung ausgeben

*************************************

ASGSTR CLR  R0             String an Basic zuweisen
       LI   R1,3
       BLWP $STRASG

RETURN MOVB $HEXFF,$BUF1   Maximale Stringlänge wiederherstellen
       MOVB $HEXFF,$BUF3   In beiden Buffern
       MOVB $HEX00,$STATUS GPL-Statusbyte löschen
       LWPI GPLWS          GPL-Workspace laden
       B    $NEXT          Sprung in den GPL-Interpreter

       END

*******************************************************************************
```

Objektcode (Dateiformat DISPLAY/FIXED 80):

```
00404STRKIT  A0000BFF00A0001A0100B00FFA0102A0202BFF00A02047F311F   0001
A0224B02E0C0204B06A0C03A4B0201B0002B06A0C0396BC060B834ABD1207F2DDF 0002
A023AC0000B0984B8101B1202B0460C03D2B06C1BD801C0000B0202C00007F316F 0003
A0250B0460C03E0B02E0C0204B06A0C03A4B0201B0002B06A0C0396BC0607F2F2F 0004
A0266B834ABD0A0C0000B0982B8042B1402B0460C03D2B6081B0204C00007F2FCF 0005
A027CBA084B06C1BD481B0460C03E0B02E0C0204B06A0C03A4B0200C00017F2C9F 0006
A0292BD060C0000B0981BD090B0982B0282B0061B1105B0282B007AB15027F302F 0007
A02A8B0222BFFE0B06C2BDC02B0601B18F2B04C0B0201B0002B0202C00007F2C3F 0008
A02BEB0420B2010B0460C03EAB02E0C0204B06A0C03A4B0200C0001BD0607F2E0F 0009
A02D4C0000B0981BD090B0982B0282B0041B1105B0282B005AB1502B02227F30DF 0010
A02EAB0020B06C2BDC02B0601B18F2B04C0B0201B0002B0202C0000B04207F2F5F 0011
A0300B2010B0460C03EAB02E0C0204B0201B0003B06A0C0396BC1A0B834A7F2E5F 0012
A0316BD220B8300B0988B0288B0005B1319BD020B8343B0980B0641B04207F2FDF 0013
A032CB200CB06A0C03B4B0581B0420B200CB0601B0420B2008B06A0C03BE7F2D2F 0014
```

```
A0342B0581B0420B2008B0580B0601B0606B18EDB0460C03EABC0C6BD0207F2C3F      0015
A0358B8343B0980B0201B0001B0202C0000BD4A0C0202B0420B2014B05817F32DF      0016
A036EB0202C0101BD4A0C0202B0420B2014B0601B0420B2010B0581B02027F330F      0017
A0384C0000B0420B2010B0580B0601B0603B18E7B0460C03EAB04C0B04207F306F      0018
A039AB200CB0420B2018B12B8B045BB04C0B0201B0001B0202C0000B04207F311F      0019
A03B0B2014B045BB0203B834AB0204C0001B1004B0203C0001B0204B834A7F31CF      0020
A03C6B0205B0004BCD33B0605B16FDB045BBD820C0202C0000B0200B1E007F2CBF      0021
A03DCB0420B2034B04C0B0201B0003B0420B2010BD820C0202C0000BD8207F311F      0022
A03F2C0202C0101BD820C0203B837CB02E0B83E0B0460B00707F502F                0023
50224LEFT$ 50254RIGHT$50286UPPER$502C6LOWER$50306SWAP  7F2D0F           0024
:       99/4 AS                                                         0025
```

Micro-Drive Sicherheitskopien – einfach erstellt! –

MD-Copierer für den ZX-Spectrum mit 48K RAM

Hierdurch wird es möglich Programme von Kassette auf Micro-Drive und umgekehrt zu kopieren.
Auch Copierer ist ein sehr nützliches Hilfsprogramm für das Microdrive: Wer wichtige Programme nicht nur einem Cartridge anvertrauen möchte, der kann damit Zweitcopien auf Kassette anfertigen, indem er das Cartridge ins MD gibt, den Kassettenrecorder auf Aufnahme laufen läßt und einfach wartet, bis alle Files des Cartridges (außer Printfiles) copiert wurden.
Umgekehrt copiert das Programm auch wieder von der Kassette zurück, dabei muß aber der Bediener das Tonbad immer wieder an- und ausschalten.
WICHTIG:
Das Programm keinesfalls vor dem Abspeichern starten, da es nicht mehr anzuhalten ist.
Tippen Sie Teil 1 ab und saven Sie diesen unter Save „COPIERER". Geben Sie Teil 2 ein und starten Sie diesen mit Run. Als Startadresse geben Sie 63650 ein. Anschließend wird der Zahlenblock eingetippt und die Prüfsumme kontrolliert. War die Prüfsumme richtig, dann geben Sie den nächsten Zahlenblock ein, war die Prüfsumme falsch, so brechen Sie das Programm mit Stop ab und starten es mit Goto 110 neu. Als Startadresse ist nun die Anfangsadresse des fehlerhaften Blocks einzugeben.

Nachdem alles vollständig eingegeben ist, speichert sich der M-Code automatisch auf Kassette ab. Nun können beide Teilprogramme auf Micro-Drive kopiert werden. Der Basic-Teil des Programms wird, wenn man es von Kassette lädt, mit Run 200 gestartet, wenn man es von Micro-Drive lädt, mit Run 300.

```
  10 REM ** COPIERER FUER MD ***
  20 PRINT "    *** COPIERER FUER
 MD ***"
  30 PRINT '" 1  CATALOGUE MIT
NULLFILES"
  40 PRINT '" 2  COPIEREN VOM B
AND AUF MD"
  50 PRINT '" 3  COPIEREN VOM M
D AUF BAND"
  60 LET A$=INKEY$: IF A$="2" TH
EN PRINT USR 63650
  70 IF A$="3" THEN PRINT USR 63
662
  80 IF A$<>"1" THEN GO TO 60
  90 PRINT USR 63681
 100 FOR X=63150 TO 63649 STEP 1
0
 110 FOR Y=0 TO 9: PRINT CHR$ PE
EK (X+Y);: NEXT Y
 120 PRINT ,: NEXT X: STOP
```

```
 200 CLEAR 50000
 210 LOAD *"M";1;"?DOPPEL MD"COD
E
 220 RUN
 300 CLEAR 50000: LOAD ""CODE :
RUN
9980 SAVE "COPIERER" LINE 300: S
AVE "?DOPPEL MD"CODE 63650,1000
9990 SAVE *"M";1;"COPIERER" LINE
 200
9995 SAVE *"M";1;"?DOPPEL MD"COD
E 63650,1000
```

```
100 CLEAR 50000
101 REM >>>>>>> Eingabe <<<<<<<
110 INPUT "Anfangsadresse ";Anf
120 FOR i=Anf TO 64640 STEP 60
130 LET Summe=0
140 FOR j=i TO i+59 STEP 6
150 PRINT j;" :";
160 FOR n=j TO j+5
170 INPUT (n;" : ");u
180 POKE n,u: LET Summe=Summe+u
190 PRINT "   "( TO 4-LEN STR$
u);u;
200 NEXT n: PRINT ""
210 NEXT j
220 PRINT '"Summe : ";Summe''
230 NEXT i
240 SAVE "?DOPPEL MD"CODE 63650
,1000
250 STOP
301 REM >>>>>>> Ausgabe <<<<<<<
310 INPUT "Anfangsadresse ";Anf
320 INPUT "Endadresse ";End
330 INPUT "Bildschirm/Drucker (
1/0) ";bd
340 IF NOT bd THEN OPEN #2,"p"
345 IF bd THEN CLOSE #2
350 FOR i=Anf TO End STEP 60
360 LET Summe=0
370 FOR j=i TO i+59 STEP 6
380 PRINT j;" :";
390 FOR n=j TO j+5
400 LET u=PEEK n
410 LET Summe=Summe+u
420 PRINT "   "( TO 4-LEN STR$
u);u;
430 NEXT n: PRINT "": NEXT j
440 PRINT '"Summe : ";Summe''
450 NEXT i: CLOSE #2
```

```
63650 :  49 174 246 243 205 104
63656 : 249 205 203 249  24 248
63662 :  49 174 246 205 198 248
63668 : 205 242 250   6 200 205
63674 :  71 252 205 165 249  24
63680 : 243 205 198 248 251 201
63686 : 243 205 212 251 205 118
63692 : 250 205   8 252  62   1
63698 : 205 247  23   1 255   0
63704 : 237  67 177 253 205 196

Summe : 10617

63710 :  18 205  83  30  32 248
63716 : 221 126  67 221 182  70
63722 : 230   2  32   5 205 254
63728 :  18  24 104 221 126  80
63734 : 183  40  98  33 174 246
63740 :  34 161 253  17  10   0
63746 :   6   0 221  78  13 121
63752 : 183  40  54 229 221 229
63758 : 197   6  10 126 221 190
63764 :  71  32   5  35 221  35

Summe : 6597

63770 :  16 245 193 221 225 225
63776 :  40  57  48   4  25  13
63782 :  24 223 229 213 197 229
63788 : 203  33  96 105   9   9
63794 :   9   9  68  77 225  43
63800 :   9 235  25 235 237 184
63806 : 193 209 225 221 229   6
63812 :  10 221 126  71 119 221
63818 :  35  35  16 247 221 225
63824 : 221 126  13  60 221 119

Summe : 7628
```

```
63830 :  13 254  50  40   6 205
63836 : 252 251 194 220 248 221
63842 : 126  13  50 163 253 201
63848 : 221  33   0  64  17  17
63854 :   0 221 229 175  55 205
63860 : 155 249 221 225  48 238
63866 :  58   0  64 254   4  48
63872 : 231 221  86  12 221  94
63878 :  11 221  33  17  64  62
63884 : 255  55 205 155 249  62

Summe : 7815

63890 :   4 211 254 216 175 211
63896 : 254  24 205  20   8  21
63902 :  62  15 211 254 195  98
63908 :   5 221  33   0  64 221
63914 : 229  17  17   0 175 205
63920 : 198   4 221 225   6  50
63926 : 205  71 252 237  91  11
63932 :  64  62 255 221  33  17
63938 :  64 205 198   4  62   4
63944 : 211 254 201 205 212 251

Summe : 7719

63950 : 221  33 168 253  42  11
63956 :  64  34 153 253  58   0
63962 :  64  50 152 253 167  32
63968 :  14  42  13  64  34 159
63974 : 253  42  15  64  34 157
63980 : 253  24  18 254   3  48
63986 :   8  58  15  64  50 157
63992 : 253  24   6  42  13  64
63998 :  34 155 253  62   1 205
64004 : 247  23 219 239 230   1

Summe : 5922
```

```
64010 : 202 227 250 205 165 250
64016 : 221 203  67 214  62   1
64022 : 205 247  23  17 250 253
64028 :  33 152 253   1   9   0
64034 : 221 113  11 237 176 213
64040 :  33   9   0 237  75 153
64046 : 253   9 203  60  36 229
64052 : 205  56  29 225 123 188
64058 : 218 227 250 209  33  17
64064 :  64 237  75 153 253 120

Summe : 8460

64070 : 177  40  33 221 126  12
64076 : 254   2  32  14 229 197
64082 : 205  13  18  62   4 211
64088 : 254 193  17 250 253 225
64094 : 237 160 221  52  11  32
64100 : 224 221  52  12  24 219
64106 : 221 203  67 206 205  13
64112 :  18  62   4 211 254 201
64118 : 221  33 168 253  33  90
64124 : 252   1  26   0 221 229

Summe : 7699

64130 : 209 237 176  33   1  64
64136 :  17 182 253   1  10   0
64142 : 237 176  17 196 253   1
64148 :  12   0  33 115 252 197
64154 : 229 237 176 225 193  17
64160 : 223 253 237 176 201 205
64166 : 118 250 205   8 252  62
64172 :   1 221 119  25 205 247
64178 :  23   1 253   2 237  67
64184 : 177 253 205 165  17  56

Summe : 8213

64190 :  22  40  17 221 203  24
64196 : 134 221 126  68 183  32
64202 :   7  33 250 253 205  70
64208 :  19 200 205 254  18 205
64214 : 252 251  32 224 221 203
64220 :  67 142 221 203  67 150
64226 : 201  33   0  64 229 209
64232 :  19  54   0   1  64  31
64238 : 237 176  24 254 205 212
64244 : 251  58 163 253 167 202

Summe : 8150

64250 : 227 250  61  50 163 253
64256 :  42 161 253  17   1  64
64262 :   1  10   0 237 176  34

64268 : 161 253 205 165 250 221
64274 : 203  24  70 194 233 251
64280 : 221 203  67  86 202 233
64286 : 251  33 250 253  17 152
64292 : 253   1   9   0 237 176
64298 :  58 152 253  50   0  64
64304 :  42 153 253  34  11  64

Summe : 8008

64310 : 167  32  14  42 159 253
64316 :  34  13  64  42 157 253
64322 :  34  15  64  24  45 254
64328 :   3  48   8  58 157 253
64334 :  50  15  64  24  33  42
64340 : 155 253  34  13  64  42
64346 : 157 253 237  91 159 253
64352 : 167 237  82  32  15 122
64358 : 187  32  11 254   6  48
64364 :   7 254   4  56   3  50

Summe : 5699

64370 :   0  64  33  17  64 237
64376 :  91 153 253 122 179 200
64382 :  34 153 253  33   8   0
64388 :  25 203  60  36 124  50
64394 : 168 253 205  13 252  17
64400 :   9   0  42 237 253 183
64406 : 237  82  34 237 253  33
64412 :   3 254 237  91 153 253
64418 :  24  22 205  27 252 221
64424 : 126  68 183  40 247  23

Summe : 7329

64430 :  61  87  30 247  42 153
64436 : 253  25 235  33 250 253
64442 : 217 205 218  18  32 228
64448 : 126 176 119 217 237  75
64454 : 237 253 237 176  58 168
64460 : 253  61  50 168 253  32
64466 : 209 201  33   0   7 227
64472 : 229  33 229 251 227 229
64478 :  33   0   0 229 195   8
64484 :   0 175 195  42  24 175

Summe : 8434

64490 : 211 254 205  42  24 205
64496 :   0   7  49 174 246  59
64502 :  59 225  43  43  43 233
64508 : 237  75 177 253  11 237
64514 :  67 177 253 120 177 201
```



```
64520 :   1 255  32  24   3   1
64526 :   0  32  33 128 252  34
64532 : 194 253 113  35  16 252
64538 : 201   1 231   4 237  67
64544 : 177 253 205 165  17  40

Summe : 7363

64550 :  14  56  12 205 218  18
64556 :  32   7  33 250 253 205
64562 :  70  19 200 205 252 251
64568 :  32 232 195 233 251  33
64574 : 168 253  17 254  91   1
64580 :   0   1 201 197 205  61
64586 : 252 235 237 176 251 193
64592 : 118  16 253 243 205  61

64598 : 252 237 176 201   0   0
64604 :   0   0 205   0   0   0

Summe : 8036

64610 :   0   0   0   0   0   0
64616 :  32  32  32  32  32  32
64622 :  32  32  32  32 255   0
64628 :   0   0   0   0   0   0
64634 :   0   0   0 255 255  65
64640 :   0   0   0   0   0   0
64646 :   0   0   0   0   0   0
64652 :   0   0   0   0   0   0
64658 :   0   0   0   0   0   0
64664 :   0   0   0   0   0   0

Summe : 1150
```

Directory-Programm für den VC-20

Dieses Programm listet die Diretory einer Diskette mit Ladegröße in Blocks und Bytes auf dem Drucker.
Man benötigt folgende Hardware: VC-20 (bei freier Speicherwahl), Diskettenlaufwerk 1541, Drucker (der Autor benutzte den MPS 801) und natürlich einen Monitor.

Laden Sie das Programm mit Load „Dir", 8 von Diskette ein und $ starten Sie mit run.
Was sonst noch zu tun ist, erklärt das Programm selbst.

```
1 REM******************
2 REM*                 *
3 REM*    D I R        *
4 REM*                 *
5 REM* WRITTEN BY:     *
6 REM* UWE PUETZ       *
7 REM* (C)  1985       *
8 REM*                 *
9 REM******************
10 PRINT"■■■":POKE36879,8
15 REM 1. SCHIRM
20 PRINT"                          ";
21 PRINT"   |PUP-SOFT PRESENTS:|   ";
22 PRINT"                          ";
23 PRINT
24 PRINT" DDD     IIIII   RRRRR";
25 PRINT" D  D      I     R   R";
26 PRINT" D   D     I     R   R";
27 PRINT" D   D     I     RRRRR";
28 PRINT" D   D     I     RR    ";
29 PRINT" D  D      I     R R   ";
30 PRINT" DDD     IIIII   R  RR";
31 PRINT
```

```
32 PRINT"                          ";
33 PRINT"   |WRITTEN BY:|          ";
36 PRINT"   |UWE PUETZ  |          ";
39 PRINT"   |(C)  1985  |          ";
40 PRINT"                          ";
41 PRINT"■■■■■TASTE DRUECKEN■ !"
50 GETA$:IFA$=""THEN50
55 REM 2. SCHIRM
60 PRINT:PRINT"■"
70 PRINT"VC-20 ALLE VERSIONEN !"
81 PRINT"■■■■ANLEITUNG■:                ";
82 PRINT" ──────                   ";
83 PRINT"■DAS PROGRAMM DIR"
84 PRINT"■IST EIN ANWENDER-"
85 PRINT"■PROGRAMM, ES DRUCKT"
86 PRINT"■DIE DIRECTORY EINER"
87 PRINT"■DISKETTE INCLUSIVE"
88 PRINT"■LADEGROESSE USW. AUS."
89 PRINT"■TASTE DRUECKEN !"
90 GETB$:IFB$=""THEN90
91 PRINT"■":REM 3. SCHIRM
92 PRINT"■BITTE LEGEN SIE EINE"
93 PRINT"■BESPIELTE DISKETTE"
94 PRINT"■INS LAUFWERK EIN,"
95 PRINT"■SCHALTEN DEN DRUCKER"
96 PRINT"■EIN UND"
97 PRINT"■■■■■DRUECKEN EINE TASTE■ !"
98 GETC$:IFC$=""THEN98
99 PRINT"■"
100 PRINT"■■■■■■■■■■■■DIRECTORY WIRD"
110 PRINT"■■■■GEDRUCKT !!!"
120 REM DISKLAUFWERK 1541 UND DRUCKER
MPS 801
130 OPEN2,8,15:OPEN4,4
140 OPEN1,8,0,"$0"
```

```
150 GET#1,A$,B$
160 GET#1,A$,B$
170 GET#1,A$,B$
180 C$="":C=0
190 IFA$<>""THENC=ASC(A$)
200 IFB$<>""THENC=C+ASC(B$)*256
210 C$=STR$(C):C$=MID$(C$,2)
220 IFLEN(C$)<2THENC$="0"+C$
230 GET#1,B$:IFST<>0THEN320
240 D$=""
250 IFB$<>CHR$(34)THEN230
260 GET#1,B$:IFB$<>CHR$(34)THEND$=D$+B$:GOTO260
270 GET#1,B$:IFB$=CHR$(32)THEN270
280 E$=""
290 E$=E$+B$:GET#1,B$:IFB$<>""THEN290
300 E$=LEFT$(E$,3)
310 IFST=0THENGOSUB330:GOTO160
320 GOTO560
330 IFI=2THEN430
340 PRINT#4,CHR$(14)CHR$(16)"20DIRECTORY":PRINT#4:PRINT#4:PRINT#4
350 Z$="GN   ARCH-NR AUFZ.-DICHTE     DOS"
360 Z$=Z$+"              ANZAHL BLOECKE   DISK-NAME"
370 PRINT#4,CHR$(15)Z$
380 PRINT#4:Z$=" 8      "+E$+"   256 BYTES/SEKT."
390 Z$=Z$+"  CBM DOS.V2.6 V170  683 (664 FREI)" +D$
400 PRINT#4,Z$:PRINT#4:PRINT#4
410 Z$="FILE - NAME       AKT.BL.     CA.BYTES              DISK-BEL.       PRG-ART"
420 PRINT#4,Z$:FORI=1TO80:PRINT#4,"-";:NEXT:PRINT#4:PRINT#4:I=2:RETURN
430 IFLEN(D$)<16THEND$=D$+" ":GOTO430
440 IFLEN(C$)<3THENC$=C$+" ":GOTO440
450 SC=SC+C:Z=Z+1
460 B=INT(C*250+.5):B$=STR$(B)
470 IFLEN(B$)<6THENB$=B$+" ":GOTO470
480 DB=INT(C/6.64*100+.5)/100:DB$=STR$(DB)
490 DB$="      "+DB$:DB$=RIGHT$(DB$,5)
500 IFMID$(DB$,LEN(DB$)-1,1)="."THENDB$=DB$+"0":GOTO490
510 IFVAL(DB$)=INT(VAL(DB$))THENDB$=DB$+".00":GOTO490
520 IFLEN(DB$)<5THENDB$=DB$+" ":GOTO520
530 Z$=D$+"  "+C$+"          "+B$+"                    "+DB$+"         "+E$
540 PRINT#4,Z$
550 RETURN
```

```
560 PRINT#4:PRINT#4
570 P=INT((664-SC)/6.64*100+.5)/100
580 B=INT(300*(664-SC))
590 PRINT#4,"FREIE BLOECKE:    ";664-SC;"IN %: "P;
600 PRINT#4,"FREIE BYTES: ";B;" EINTRAEGE: ";Z
610 PRINT#4,"FREIE EINTRAEGE: "144-Z
620 PRINT#4:PRINT#4
630 CLOSE1:CLOSE2:CLOSE3:CLOSE4
650 PRINT"■"
700 PRINT"■■■■■■■■■■■■■■■E N D E  !"
```

DIRECTORY PROBELAUF DEMO-DISKETTE

GN	ARCH-NR	AUFZ.-DICHTE	DOS	ANZAHL BLOECKE	DISK-NAME
8	11	256 BYTES/SEKT.	CBM DOS.V2.6 V170	683 (664 FREI)	DEMO-DISK

FILE - NAME	AKT.BL.	CA.BYTES	DISK-BEL.	PRG-ART
PERFORMANCE TEST	09	2250	1.36	PRG
HOW TO USE	13	3250	1.96	PRG
HOW PART TWO	05	1250	.75	PRG
VIC-20 WEDGE	04	1000	.60	PRG
DOS 5.1	04	1000	.60	PRG
COPY/ALL	11	2750	1.66	PRG
PRINTER TEST	09	2250	1.36	PRG
DISK ADDR CHANGE	04	1000	.60	PRG
DIR	04	1000	.60	PRG
VIEW BAM	06	1500	.90	PRG
CHECK DISK	04	1000	.60	PRG
DISPLAY T&S	14	3500	2.11	PRG
SEQUENTIAL FILE	05	1250	.75	PRG
RANDOM FILE	13	3250	1.96	PRG

FREIE BLOECKE: 559 IN %: 84.19 FREIE BYTES: 167700 EINTRAEGE: 14
FREIE EINTRAEGE: 130

Was nutzen die besten Hi-Res Grafiken, wenn man nicht in der Lage ist, sie auf Papier zu bannen. Dies erledigt das abgedruckte Programm Hardcopy in knapp drei Minuten.

Hardcopy-Routine für den C64 + MPS801

Leider ist diese Routine wegen der vielen verschiedenen Druckertypen nur für den C64 in Verbindung mit einem Commodore MPS801 oder einem kompatiblen Drucker (z.B. VC1515,GP100VC...) geeignet. Abgedruckt ist der Basic-Loader, obwohl das Programm selbst aus Geschwindigkeitsgründen vollständig in Maschinensprache geschrieben ist (ein entsprechendes Basic-Programm würde vermutlich mehrere Stunden benötigen!); es empfiehlt sich natürlich, das Programm als reines Maschinenprogramm abzuspeichern, da es in dieser Form um einiges kürzer und deshalb schneller zu laden ist.

Dazu enthält der Basic-Loader schon eine kurze Save-Routine. Starten Sie also den Basic-Loader, nachdem Sie ihn abgetippt und zur Sicherheit abgespeichert haben, mit RUN! Haben Sie beim Eingeben der Data-Werte einen Fehler gemacht, meldet das Programm Ihnen dies und Sie müssen ihn vor neuerlichem Start berichtigen. Ist alles korrekt eingegeben, fragt das Programm, ob sie auf Diskette oder auf Kassette abspeichern möchten; geben Sie entsprechend „D“ oder „K“ ein und drücken Sie, wenn Sie eine Diskette/Kassette vorbereitet haben, die SPACE-Taste; schon wird das Programm abgesaved. Es ist wirklich erstaunlich kurz (auf Diskette ein Block).

Das Programm kann dann jederzeit mit LOAD“HARDCOPY.CODE“,8,1 (Datassetten-Anwender müssen die 8 natürlich durch eine 1 ersetzen) nachgeladen und mit SYS49152 ohne vorheriges Einlesen der Data-Werte gestartet werden.

Es erzeugt dann auf dem Drukker die Hardcopy einer Hi-Res-Grafik, die in dem Bereich von dez.8192-16191 (hex.2000-3fff) abgelegt sein muß; gegebenenfalls muß eine Verschiebung in diesen Bereich vorgenommen werden, oft liegt hier aber schon der Grafikbildschirm, so z.B. die Hires-Grafik auf der Commodore-Demo-Diskette (Frauenkopf). Zu beachten ist noch, daß das Programm die logische Filenummer 127 verwendet.

Änderungsvorschlag:
Manchmal ist es sinnvoll, die Grafik zu invertieren, bevor man sie ausdruckt (das ist z.B. auch bei der Grafik-Demo auf der Commodore-Demo-Diskette der Fall); diese Möglichkeit ist optional in die Routine eingebaut worden. Dazu sind nur folgende Änderungen am Basic-Loader vorzunehmen:

- In Zeile 115 sind das 11. und das 12. Datum (..,234,234,..) zu ersetzen durch “73,127“.
- Entsprechend muß natürlich die Prüfsumme in Zeile 150 von 24481 in 24213 geändert werden.

Ansonsten gelten für den Basic-Loader, dann dieselben Bedienungshinweise wie sonst üblich.

```
0 REM *** HARDCOPY-ROUTINE FUER DEN ***
1 REM *** C64 IN VERBINDUNG MIT DEM ***
2 REM *** MPS-801 ODER EINEM KOMPA- ***
3 REM *** TIBLEN DRUCKER (GP100VC)! ***
4 REM ***                           ***
5 REM *** GRAFIK IN HEX.$2000-$3FFF ***
6 REM ***                           ***
7 REM *** WRITTEN 06/85 BY          ***
8 REM ***                           ***
9 REM ***          WALDEMAR RAAZ    ***
```

```
10 :
11 :
100 REM MASCHINEN-ROUTINE
101 :
110 DATA169,127,133,184,169,4,133,186,169,0,133,183,32,192,255,162,127,32
111 DATA201,255,169,13,32,210,255,169,8,32,210,255,169,192,133,2,169,30,133
112 DATA3,169,0,133,4,169,32,133,5,169,28,133,8,169,7,133,6,169,40,133,7
113 DATA164,6,192,7,240,20,162,7,177,2,10,126,60,3,202,16,249,200,192,8,208
114 DATA240,165,6,240,18,160,0,162,7,177,4,10,126,60,3,202,16,249,200,196
115 DATA6,208,240,162,7,189,60,3,56,106,234,234,164,8,208,2,41,143,32,210
116 DATA255,202,16,237,24,169,8,101,2,133,2,144,2,230,3,169,8,24,101,4,133
117 DATA4,144,2,230,5,198,7,208,165,169,13,32,210,255,198,8,208,7,169,3,133
118 DATA6,76,54,192,16,18,169,15,32,210,255,169,13,32,210,255,32,204,255
119 DATA169,127,76,195,255,198,6,48,3,76,54,192,165,2,133,4,165,3,133,5,56
120 DATA165,2,233,64,133,2,165,3,233,1,133,3,76,50,192
121 DATA72,65,82,68,67,79,80,89,46,67,79,68,69
125 :
126 :
127 REM EINLESEN DER MA-ROUTINE
128 :
130 RESTORE:SU=0:PRINTCHR$(147)
140 FORI=49152TO49380:READA:POKEI,A:SU=SU+A:NEXT
150 IFSU<>24481THENPRINT" FEHLER IN DATAS !!!":STOP
151 :
152 :
160 REM ABSPEICHERN DER MA-ROUTINE
161 :
170 PRINT"■ UM DIE HARDCOPY-ROUTINE SCHNELLER LA-"
180 PRINT" DEN ZU KOENNEN, SPEICHERT MAN SIE AM"
190 PRINT" BESTEN ALS REINES MASCHINENPROGRAMM"
200 PRINT" AB. WOLLEN SIE AUF ■D■ISKETTE ODER"
210 PRINT" ■K■ASSETTE SPEICHERN ?"
220 GETG$:IFG$<>"K"ANDG$<>"D"THEN220
230 GA=8:IFG$="K"THENGA=1
240 PRINT"■ DRUECKEN SIE DIE SPACE-TASTE, WENN"
250 PRINT" EINE DISKETTE/KASSETTE EINGELEGT IST!"
255 GETG$:IFG$<>" "THEN255
260 POKE185,1:POKE186,GA:POKE187,216:POKE188,192:POKE183,13
270 POKE251,0:POKE252,192:POKE780,251:POKE781,216:POKE782,192:SYS65496
```

AUTONUMBER-Utility-Routine C-64

Besonders bei längeren Programmen verschwendet man einige Zeit allein mit dem lästigen Eintippen der Zeilennummern. Das Maschinenprogramm AUTONUMBER gibt jede neue Zeilennummer, mit einer von Ihnen gewählten Schrittweite, automatisch aus.

Geben Sie zunächst den Basic-Loader ein und starten Sie ihn. Nach kurzer Wartezeit meldet sich das Basic-Programm mit der Frage, ob Sie das reine Maschinenprogramm auf Diskette oder auf Kassette abspeichern wollen. Benötigen Sie es, so laden Sie es mit LOAD „AUTONUMBER",8,1 (von Kassette mit ,1,1) vor dem Eingeben des Basic-Programmes und geben Sie NEW ein (um nach dem Laden des Maschinenprogrammes die Zeiger für Variablen-Anfang usw. wieder zurückzusetzen). Mit SYS 49245, SW aktivieren Sie nun die automatische Zeilenausgabe, wobei SW die Schrittweite ist. SW kann Werte von 0 bis 63999 haben. Die erste Zeilennummer muß noch „per Hand" eingegeben werden, um die Start-Zeilennummer festzulegen.

Drücken Sie nach Eingabe einer Basic-Zeile jetzt RETURN, erscheint in der nächsten Zeile automatisch die neue Zeilennummer mit dem Abstand, der von Ihnen eingegebenen Schrittweite. Überschreitet die Zeilennumerierung den Wert 63999 (höchste zulässige Zeilennummer), so wird die zuletzt eingegebene Zeile ignoriert und ein ILLEGAL QUANTITY ERROR ausgegeben. Drükken Sie ohne Eingabe von Basic-Text nur RETURN, oder löschen Sie die Zeilennummer und geben etwas anderes ein, so wird die automatische Zeilennumerierung abgebrochen. Bis Sie sie wieder mit SYS 49245, SW neu aktivieren, können Sie jetzt wieder Einzelzeilen eingeben.

Beispiel:

SYS 49245,10	legt die Schrittweite 10 fest
50 A=5	Eingabe der ersten Zeilennummer
60	wird automatisch ausgegeben


```
10 REM ********************************
15 REM *                              *
20 REM *      AUTONUMBER-UTILITY      *
25 REM *                              *
30 REM * EINGABEFORM: SYS 49245,SW    *
35 REM *                              *
40 REM * 1985 BY DANIEL DURSTEWITZ    *
45 REM *         REICHENBERGER STR.26 *
50 REM *         4460 NORDHORN        *
55 REM *         TEL.05921/79200      *
60 REM *                              *
65 REM ********************************
70 :
75 :
80 FORT=49152TO49306:READA:POKET,A:P=P+A:NEXT
85 IFP=20338GOTO100
90 PRINT"■FEHLER IN DEN DATAS! BITTE UEBERPRUEFEN"
95 PRINT"SIE DIE ZEILEN 136-151 NOCH EINMAL!":END
100 PRINT"■SOLL DAS PROGRAMM AUF ■D■ISKETTE ODER"
105 INPUT"AUF ■K■ASSETTE ABGESPEICHERT WERDEN";A$
110 IFA$="D"THENPOKE49274,8:GOTO125
115 IFA$="K"THENPOKE49274,1:GOTO125
120 GOTO100
125 SYS49273:END
130 :
135 :
136 DATA32,96,165,134,122,132,123,32,115,0,170,240,243,162,255,134,58,144
137 DATA13,169,131,141,2,3,169,164,141,3,3,76,150,164,32,107,169,240,238
138 DATA165,20,164,21,24,101,251,170,144,3,200,240,35,152,24,101,252,176
139 DATA29,32,205,189,162,255,232,189,0,1,157,119,2,208,247,169,32,157,119
140 DATA2,232,134,198,169,0,133,211,76,159,164,76,72,178,234,234,234,234
141 DATA234,32,253,174,32,138,173,32,247,183,165,20,133,251,165,21,133,252
142 DATA169,0,141,2,3,169,192,141,3,3,96
145 :
146 REM  SAVE-ROUTINE
147 :
150 DATA162,1,160,1,32,186,255,169,10,162,92,160,8,32,189,255,169,0,133,251
151 DATA169,192,133,252,169,251,162,121,160,192,32,221,245,96
```

Assemblerlisting

```
C*
      PC   IRQ   NV-BDIZC AC XR YR SP
.;  E147 EA31  10110000 C0 61 40 F8
.
.,  C000  20 60 A5   JSR $A560
.,  C003  86 7A      STX $7A
.,  C005  84 7B      STY $7B
.,  C007  20 73 00   JSR $0073
.,  C00A  AA         TAX
.,  C00B  F0 F3      BEQ $C000
.,  C00D  A2 FF      LDX #$FF
.,  C00F  86 3A      STX $3A
.,  C011  90 0D      BCC $C020
.,  C013  A9 83      LDA #$83
.,  C015  8D 02 03   STA $0302
.,  C018  A9 A4      LDA #$A4
.,  C01A  8D 03 03   STA $0303
.,  C01D  4C 96 A4   JMP $A496
.,  C020  20 6B A9   JSR $A96B
.,  C023  F0 EE      BEQ $C013
.,  C025  A5 14      LDA $14
.,  C027  A4 15      LDY $15
.,  C029  18         CLC
.,  C02A  65 FB      ADC $FB
.,  C02C  AA         TAX
.,  C02D  90 03      BCC $C032
.,  C02F  C8         INY
.,  C030  F0 23      BEQ $C055
.,  C032  98         TYA
.,  C033  18         CLC
.,  C034  65 FC      ADC $FC
.,  C036  B0 1D      BCS $C055
.,  C038  20 CD BD   JSR $BDCD
.,  C03B  A2 FF      LDX #$FF
```

```
.;  C03D  E8          INX
.;  C03E  BD 00 01    LDA $0100,X
.;  C041  9D 77 02    STA $0277,X
.;  C044  D0 F7       BNE $C03D
.;  C046  A9 20       LDA #$20
.;  C048  9D 77 02    STA $0277,X
.;  C04B  E8          INX
.;  C04C  86 C6       STX $C6
.;  C04E  A9 00       LDA #$00
.;  C050  85 D3       STA $D3
.;  C052  4C 9F A4    JMP $A49F
.;  C055  4C 48 B2    JMP $B248
.;  C058  EA          NOP
.;  C059  EA          NOP
.;  C05A  EA          NOP
.;  C05B  EA          NOP
.;  C05C  EA          NOP
.;  C05D  20 FD AE    JSR $AEFD
.;  C060  20 8A AD    JSR $AD8A
.;  C063  20 F7 B7    JSR $B7F7
.;  C066  A5 14       LDA $14
.;  C068  85 FB       STA $FB
.;  C06A  A5 15       LDA $15
.;  C06C  85 FC       STA $FC
.;  C06E  A9 00       LDA #$00
.;  C070  8D 02 03    STA $0302
.;  C073  A9 C0       LDA #$C0
.;  C075  8D 03 03    STA $0303
.;  C078  60          RTS
.;  C079  EA          NOP
.;  C07A  EA          NOP
.;  C07B  EA          NOP
.;  C07C  EA          NOP
.;  C07D  EA          NOP
```

CPU-Hardware-Test: Matrixdrucker „Star SG-10“

Vor einigen Monaten wurde der Nachfolger des erfolgreichen Matrixdruckers Gemini 10 X vorgestellt. STAR präsentierte zusammen mit einer ganzen Serie neuer Druckertypen den SG-10, der einige außergewöhnliche Leistungsmerkmale für seine Preisklasse besitzt.

Der Matrixdrucker ist serienmäßig sowohl für Einzelblatt-Druck als auch für Endlospapier geeignet, da der Traktor schon im Lieferumfang enthalten ist, was ja keineswegs selbstverständlich ist.

Allerdings ist mit der Benutzung des Traktors ein Nachteil verbunden: Da dieser sich nicht (wie z. B. bei Epson FX-80) auf Höhe der Walze und des Farbbandes befindet, sondern etwa 5 cm höher, geht nach dem Einspannen des Endlospapiers vom ersten Blatt stets ein großes Stück am oberen Rand verloren.

Der SG-10, der wie sein Vorgänger ein Rollenfarbband verwendet, ist mit einem 2k-Byte großen Speicher ausgestattet, der neben der Pufferfunktion auch benutzerdefinierte Zeichen speichern kann. Dieser Puffer kann übrigens auf maximal 10 kByte erweitert werden.

Auch der Ausdruck in kursiver Schrift ist selbstverständlich möglich.

Bei den normalen ASCII-Zeichen hat der SG-10 eine Matrixauflösung von 9 * 11 Punkten, im NLQ-Modus beträgt die Auflösung immerhin 17 * 11 Punkte.

Der SG-10 erreicht im Normal-Druck eine Geschwindigkeit von 120 Zeichen/Sekunde. In speziellen Druckmodi (z. B. Doppeldruck, NLQ) ist die Geschwindigkeit natürlich nicht so hoch. Normalerweise druckt der STAR-Drucker bidirektional, was sich jedoch über Steuerzeichen ein- oder ausschalten läßt. Außerdem besitzt er eine Druckwegoptimierung.

Es ist natürlich keine Frage, daß der STAR SG-10 auch grafikfähig ist. Zunächst besitzt er eine Reihe von Blockgrafik-Zeichen, außerdem ist über Einzelnadelansteuerung natürlich auch hochauflösender Grafikdruck möglich. Beim Anschluß des SG-10 an den Apple II mittels des Grafstar-Interfaces ist beispielsweise eine Hardcopy von einer der Grafikseiten des Apple II kein Problem.

Mit dem Befehl PRINT CHR$(9);"G" wird die erste Grafikseite ausgedruckt. Es gibt auch noch weitere Parameter, die das Vergrößern, Invertieren, Drehen sogar das logische Verknüpfen beider HGR-Seiten erlaubt.

Ein besonderes Steuerzeichen stellt der sogenannte Macro-Steuerbefehl dar. Man kann eine große Zahl von häufig zusammen benutzten Steuerzeichen mit dem Macro-Code zusammenfassen und über ein Steuerzeichen aufrufen. Die Definition dieses Macro-Codes ist sehr einfach und auch im übrigens deutschsprachigen 251 Seiten starken Handbuch gut erklärt. Dieses Manual ist wirklich nachahmenswert gestaltet; neben gut verständlichen Erklärungen existieren eine Vielzahl von Tabellen, denen man beim Suchen nach einer Information alles Wichtige entnehmen kann.

Serienmäßig ist der SG-10 mit einer Centronics Parallelschnittstelle ausgestattet, kann jedoch auch mit einer seriellen Schnittstelle ausgerüstet werden.

Bei einem Preis von 1195 DM (unverbindliche Preisempfehlung) ist das Gerät meiner Ansicht nach der zur Zeit beste Matrixdrucker seiner Preisklasse, zumal er sich nicht nur an die gängigen Homecomputer anschließen läßt, sondern mit einem eigenen Zeichen- und Befehlssatz auch für den Anschluß an IBM-Geräte vorbereitet ist. Der Drucker ist übrigens bei vielen Händlern zu einem noch weit niedrigeren Preis erhältlich, der Preisvergleich lohnt sich also.

Viele Funktionen des Druckers (so zum Beispiel die nationalen Zeichensätze) können sowohl über DIP-Switches als auch mittels Steuerzeichen eingestellt werden. Die DIP-Switches befinden sich an der linken Seite des Druckers und sind somit stets leicht zugänglich, ohne daß das Gehäuse geöffnet werden muß.

Der STAR SG-10 kann Texte in Pica, Elit, Schmalschrift, Proportionaldruck, Breitdruck, Fettdruck, Doppeldruck, Superscript, Subscript und NLQ ausdrucken. NLQ steht dabei für NEAR LETTER QUALITY. Von der Qualität dieser Schönschrift, die einem Typenraddrucker gleichkommt ist wirklich überzeugend.

Oliver Steinmeier

Die neuen 16-Bit-Rechner im Vergleich

Drei der Superrechner sind schon sehr bekannt. Es handelt sich um den **Macintosh von Apple**, den **Atari 520 ST** und den **Sinclair QL**. Der vierte Rechner weist eine Besonderheit auf: er wurde in Deutschland entwickelt und wird auch hier gefertigt. Es handelt sich um den bei vielen Apple II-Freaks schon bekannten **GEPARD**.

Was bieten die einzelnen Systeme?

Als der Mac von Apple der Öffentlichkeit vorgestellt wurde, sprach man von einer neuen Generation der Bürocomputer. Das lag hauptsächlich an dem revolutionären Konzept, das aus Maussteuerung und Fenstertechnik mit Symbolen besteht. Zwar besaß schon die ein Jahr zuvor präsentierte Lisa von Apple diese Technik, doch wer konnte sich schon ein Computersystem für etwa 30 000 DM leisten. Der Mac hingegen kostete nur etwa ein Drittel des Lisa-Preises und war damit schon erschwinglicher. Der Mac, der den MC68000-Prozessor in der 8-Mhz-Version enthält, bietet 128 kByte RAM. Seit einiger Zeit gibt es jedoch auch den sogenannten Fat-Mac, der von Beginn an schon einen RAM-Speicher von 512 kBte besitzt. Dieser Speicherzuwachs wurde gebraucht, da die aufwendigen Macintosh-Programme immer mehr Speicher benötigten und für Daten somit nur eine geringe Kapazität vorhanden war. Der 128k-Mac läßt sich übrigens für etwa 1 400 DM auf 512 kByte aufrüsten.

In den 64k ROM sind viele Routinen zur Unterstützung der aufwendigen Bildschirmdarstellung enthalten.

Als Massenspeicher bietet Apple neben dem eingebauten 3 1/2 Zoll Diskettenlaufwerk mit einer Kapazität von 400 kByte eine zweite externe Floppy gleichen Formats an. Außerdem gibt es demnächst eine 20 Mbyte-Festplatte für den Mac.

Die Grafik des Mac hat eine Auflösung von 512 * 342 Punkten, jedoch ist leider keine Farbgrafik möglich.

Etwas einfach wirkt die Mac-Tastatur, die nicht einmal einen 10er-Block aufweisen kann. Allerdings ist die Tastatur beim Macintosh nach der Meinung des Herstellers zumindest teilweise durch die Maus ersetzt worden.

Bei der Einführung des Macintosh wurde eine Flut von Software für dieses System angekündigt. Zwar gibt es mittlerweile neben vielen professionellen Softwarepaketen auch einige Spiele (z. B. Zork, Transylvania), doch von einer regelrechten Softwareflut wie z. B. für den Apple II oder den C-64 kann man noch nicht sprechen. Vor einiger Zeit wurde Clive Sinclairs neues Wunderding, der Sinclair QL groß angekündigt. QL soll für Quatum Leap (Quantensprung) stehen und den technischen Vorsprung dieses Rechners verdeutlichen. Doch trotzdem war dieses System bei uns bisher kein großer Verkaufserfolgt. Das mag auch teilweise an der Firmenpolitik des britischen Herstellers gelegen haben, der mit der Markteinführung lange wartete, obwohl das Gerät schon weitaus früher angekündigt wurde. Außerdem begann man mit der Auslieferung der deutschen Version sehr spät, so daß viele potentielle Käufer verärgert auf ein anderes System (z. B. Schneider oder C-64) umstiegen. Ein weiteres Handicap für den QL liegt wohl in dem geringen Softwareangebot, das bisher für den Rechner vorhanden ist.

Doch nun zu den technischen Details: Der QL enthält nach Sinclairs Datenblatt den 32-Bit-Mikroprozessor 68008. Zwar bietet dieser zur 68000-Familie gehörende Prozessor intern 32-Bit-Register, der Datenbus, über den z. B. mit dem RAM kommuniziert wird, hat jedoch nur eine Breite von 8 Bit. Deshalb läßt sich auch darüber streiten, ob der QL ein echter 32-Bitter ist. Im Gegensatz zum 68008 hat der 68000 (z. B. im Mac) ein 16-Bit-Datenregister und ist somit im Vergleich zum kleineren Bruder 68008 schneller. Der Prozessor im QL hat keine Taktfrequenz von 7,5 Mhz und wird durch einen zweiten Prozessor (Intel 8049) entlastet, der für Tastatur, serielle Schnittstelle und Sound verantwortlich ist. Die RAM-Kapazität des Sinclairs beträgt in der Grundausstattung 128 kByte und kann auf 640 k erweitert werden. Die Tastatur besitzt 5 Funktionstasten und ist bei der deutschen Version des QL in der QWERTZ-Form mit Umlauten ausgestattet. Als Massenspeicher verwendet der QL, die vom Sinclair Spectrum her bekannten Microdrives von denen zwei mit einer Kapazität von je etwa 100 kByte eingebaut sind. Diese Laufwerke sind der Hauptkritikpunkt, wenn man den QL mit dem Anspruch eines Personalcomputers verkauft. So ist z. B. die Kapazität einer Microdive Cartridge viel zu gering für professionelle Nutzung. Doch mittlerweile sind schon mehrere Diskettenlaufwerke von Fremdherstellern verfügbar, so daß das Massenspeicherproblem nicht mehr so von Bedeutung ist.

Der QL hat eine Grafikauflösung von 512 * 256 Pixels (Bildpunkten), wenn man sich auf vier Farben beschränkt. Bei Verwendung von 8 Farben ist eine Auflösung von 256 * 256 Punkten möglich.

Neben zwei Joysticks bietet der QL zwei RS-232-Schnittstellen (seriell) und einen Netzwerkanschluß für ein eigenes Netzwerk, mit dem maximal 64 QLs zusammengeschlossen werden können. Als Bildschirmgerät kann neben einem Monitor auch ein Fernseher verwendet werden.

Beim Kauf des Ql erhält man zu der Hardware noch die vier Softwarepakete QUILL (Textverarbeitung), ABAKUS (Tabellenkalkulation), ARCHIVE (Datenbank) und EASEL (Businessgrafik) dazu, die eine recht gute Qualität haben. Die deutsche Version des QL soll etwa 1 700 DM kosten, während die Geräte mit englischer Tastatur im Augenblick schon etwa 200 DM billiger sind.

Über den Atari 520 ST braucht eigentlich nicht viel gesagt werden. Dieser Rechner ist seit der Hannovermesse wohl in (fast) aller Munde. Für einen Preis von unter 3 000 DM bietet Jack Tramiel, der ehemalige Commodore-Chef und jetzige Atari-Oberste ein Gerät, das dem Macintosh von Apple den Wind aus den (Verkaufs-) Segeln nehmen soll. Wie der Mac arbeitet der 520 ST mit der 8-Mhz-Version des 68000. Schon in der Grundausstattung bietet er jedoch 512 kByte RAM, was praktisch einer Sensation gleichkommt. Die 192 kByte ROM, die neben Basic, Logo auch noch GEM enthalten sollen, sind noch nicht „das letzte Wort“ des Atari: es kann sogar noch auf 340 kByte erweitert werden. Das GEM-Betriebssystem erlaubt dem Atari, eine dem Macintosh nachempfundene Benutzeroberfläche mit Maus, Window und Icons. Wie der Mac bietet Atari ein 3 1/2-Zoll-Laufwerk an. Hier wird wieder einmal deutlich, daß die Tendenz immer mehr zu den kleinen Laufwerken geht. Im Preis von unter 3 000 DM ist ein Laufwerk enthalten, was jedoch nicht im Gehäuse eingebaut ist,

sondern extern angeschlossen wird. Es soll eine Kapazität von 500 kByte haben. Außerdem hat der ST-520 noch einen Festplattenanschluß serienmäßig eingebaut, an den die 10 Mbyte-Harddisk, die Atari für einen sensationell niedrigen Preis anbieten will, angeschlossen werden soll.

Die Grafik des Jackintosh, wie der 520 ST inzwischen in Anlehnung an Jack Tramiel und Macintosh genannt wird, hat eine maximale Auflösung von 640 * 400 Bildpunkten, wobei dann „nur" eine Schwarzweiß-Darstellung möglich ist. Bei etwas niedrigerer Auflösung kann man jedoch aus der großen Farbenvielfalt von 512 Farben wählen. Bei 640 * 200 Punkten können 4 Farben dargestellt werden, bei 320 * 200 Punkten sogar 16 Farben gleichzeitig. Hier hat der Jackintosh also einen eindeutigen Vorteil gegenüber dem Macintosh.

Auch die Tastatur macht zumindest vom äußeren Aussehen einen besseren Eindruck als die des Mac. Im Gegensatz zum Apple-Rechner sind 10er-Block, Cursor-Block, 10 Funktionstasten vorhanden. Das Tastengefühl jedoch ist Ansichtssache, so daß darüber hier nicht geurteilt werden soll.

Wie bei der Einführung des Mac wird eine große Softwareflut erwartet. Was daraus wird, werden wir in den nächsten Monaten erleben. Von Vorteil für den Atari wird jedoch das Betriebssystem TOS (Tramiel Operating System) sein, daß aus dem CP/M 68k entwickelt wurde. Auf Grund dieser Verwandtschaft sollte eine Übertragung von Programmen eigentlich relativ einfach sein.

Nun kommen wir zum GEPARD, einem wie oben schon erwähnt, in Deutschland entwickelten 68000-System, daß in den Kreisen der Apple-Freaks schon einiges Aufsehen erregt hat. Im Gegensatz zu den bisher vorgestellten Rechnern ist der Gepard ein vollständig modular aufgebautes System, was wohl etwas an den guten, alten Apple II erinnert. Zu Recht, würden die Entwickler des Gepard wohl sagen, denn der Gepard wurde von Apple-Freaks entwickelt, denen der 6502-Rechner zu klein und zu langsam wurde. Da sie schon einige Erfahrung im Bau von Apple-Pheripherie hatten, wagten sie das Risiko und gründeten eine Firma, die Gepard Computer GmbH mit Sitz in Oldenburg, um ihren 68000-Rechner verkaufen zu können.

Der Gepard besitzt eine 68000-CPU mit einer Taktfrequenz von 10 Mhz, es ist jedoch auch eine 12-Mhz-CPU erhältlich.

Die CPU ist auf einer der Steckkarten (Europaformat) untergebracht, die in den Slots des Gepard sitzen. Somit ist es nach Angaben des Herstellers durch Austausch der Prozessorplatine möglich, den 68020 einzusetzen, sobald dieser lieferbar ist. Der 68020 ist der große Bruder des 68000 und besitzt einen 32-Bit Datenbus und kann somit mit Fug und Recht als 32-Bit-Prozessor bezeichnet werden. Insgesamt sind 16 Steckplätze vorhanden, in denen neben der CPU-Karte auch eine I/0-Karte, RAM-Karten, Floppy-Controller, parallele- und serielle Interfaces, Videokarte und Grafikkarte untergebracht werden können. Der Floppy-Controller kann 5 1/4 Zoll und 3 1/2 Zoll Laufwerke ansteuern. Der Hersteller empfiehlt die kleineren 3 1/2 Zoll-Laufwerke von Sony mit einer formatierten Kapazität von 800 kByte.

Die RAM-Karten gibt es in den Größen 128 kByte und 512 kByte. Über die I/0-Card kann der Gepard an andere Rechner angeschlossen werden. Mit Hilfe eines speziellen Interfaces und der I/0-Card kann der Gepard den Apple als Ein-/Ausgabestation benutzen. Damit ist es mit einer Minimalausrüstung möglich, diesen Rechner zu betreiben: Man benötigt nur den Apple II mit Laufwerken, 80-Zeichenkarte und UCSD-Pascal. Der Gepard nutzt dann die Apple-Tastatur, die Apple-Floppies und den Bildschirm über die Apple-Ausgabe.

Das nötige Gepard-System kostet dann nur knapp 2000 DM. In naher Zukunft soll es für den C-64 ebenfalls ein entsprechendes Interface geben.

Selbstverständlich ist es auch möglich, den Gepard als eigenständiges System zu betreiben. Für knapp 7 400 DM erhält man ein Gehäuse mit Lüfter, Lautsprecher und einem Laufwerk. Ebenfalls enthalten sind die Busplatine mit den 16 Steckplätzen sowie eine CPU-Platine, eine RAM-Karte mit 512 Kbyte, eine I/0-Platine, ein Floppycontroller, eine Videokarte, eine Grafikkarte mit Grafikprozessor und 128 kByte Video-Ram. Die Farb-Grafik des Gepard hat eine maximale Auflösung von 640 * 480 Pixels. Mit dem Gepard-System wird selbstverständlich auch Software mitgeliefert. Der Käufer erhält kostenlos einen vollständigen Modula-II-Compiler, der vollständig in 68000-Assembler geschrieben wurde, einen symbolischen Assembler, einen Editor und einen Monitor mit Disassembler.

Auf dem Gepard soll mit CP/M 68k in Zukunft auch ein Standardbetriebssystem laufen, um die Übertragung von Programmen zu erleichtern.

Der Gepard wird mit einem ausführlichen Handbuch in deutscher Sprache ausgeliefert. Die Firma Gepard Computer bietet einen nachahmenswerten Service für ihre Kunden: Es existiert ein Update-Service. Sobald eine neue Version der Software existiert oder das Handbuch ergänzt wird, erhält der Gepard-Besitzer kostenlos eine Update-Sendung.

Zwar wird es für den Gepard niemals ein besonders großes kommerzielles Softwareangebot für den Büroeinsatz geben, da der Verbreitungskreis des Rechners einfach zu klein ist. Eine Firma wie die Gepard Computer GmbH kann sich keinen Werbeetat leisten, der auch nur einem Zehntel des Etats von Firmen wie Apple, Atari oder Commodore entspricht. Für Freaks, Techniker und Hochschulen erscheint der Rechner jedoch hervorragend geeignet, um eigene Projekte zu verwirklichen.

Der Macintosh hingegen ist nicht so sehr für Freaks und Hobbyprogrammierer geeignet, sondern wird vielmehr seine Verbreitung in Büros und somit bei professionellen Anwendungen finden.

DerSinclair QL wird dagegen kaum als professioneller Rechner in Büros eingesetzt werden. Vielmehr wird wohl die Mehrzahl der Geräte bei uns und vor allem in England im Heimbereich abgesetzt werden.

Technisch gesehen sollte es dem Atari 520 ST nicht schwerfallen, ebenfalls im professionellen Bereich wie der Mac von Apple eingesetzt zu werden. Dagegen spricht allerdings das Image, daß die Firma Atari hat. Bei vielen potentiellen Käufern hat dieses Unternehmen den Ruf eines Herstellers von Videospielen und Homecomputern. Deshalb wird ein Unternehmen eher den Macintosh von Apple als den 520 ST von Atari anschaffen. Für Firmen ist der Preis dann weniger bedeutsam.

Deshalb ist der Atari 520 ST als Homecomputer der obersten Preiskategorie einzuordnen, wobei der Erfolg dieses Rechners wohl entscheidend vom Softwareangebot abhängen wird.

Martin Kotulla

* * * *

Spectrum-Disassembler

Zuerst einige Vorzüge dieses Disassemblers:

1. Dieser Disassembler ist vollständig in Maschinensprache geschrieben

Daraus ergeben sich als Konsequenz hohe Schnelligkeit und relativ geringer Speicherplatzbedarf.

2. Dieser Disassembler ist voll relocable

Dadurch können Assemblerprogramme in dem Speicherbereich untersucht werden, für den sie geschrieben wurden. Lästige Umrechnungen bei JP's und CALL's entfallen.

3. Der Disassembler ist vollständig

Alle ZILOG-Befehle (sogar einer mehr: SLA! ≙ arithm. Linksshift mit Nachreichen einer 1) werden übersetzt, insbesondere auch Index-Befehle mit OFFSET-Angabe. Bei relativen Sprungbefehlen wird zusätzlich die absolute Adresse angegeben.

4. Der Disassembler besitzt ein hexadezimales Zahlendarstellungsformat

Durch Drücken einer Taste (H) kann jederzeit der Darstellungsmodus umgeschaltet werden.

5. Das Programm erstellt ein korrektes ROM-Disassembly

- Listen im ROM werden nicht assembliert, sondern mit DEFB angegeben.
- Nach einem RSTØ8 (Error-Restart) wird das nachfolgende Byte, das zur Fehleridentifikation dient, ebenfalls mit DEFB angegeben.
- Die Pseudo-Sprache des Sinclair-ROM-FP-Calculators wird übersetzt (der FP-Calculator wird durch „RST 28" aufgerufen).

6. Dieser Disassembler besitzt einen MEMORY EDITOR

Der Speicherinhalt wird sowohl zahlenmäßig, als auch als ASCII-Charakter-Folge dargestellt (sehr hilfreich, falls seltsames Disassembly auf Datenbereich schließen läßt).

7. Dieser Disassembler besitzt zusätzlich einen Hex-Dez-Loader

Dadurch können beim Untersuchen von Programmen sofort kleinere Veränderungen vorgenommen werden. Außerdem kann der Hex-Dez-Loader zur Eingabe von MC-Listings verwendet werden, da eine Prüfsumme mitgeführt wird (Nullsetzen der Prüfsumme mit „Z").

8. Der Disassembler besitzt vielseitige Steuerungsmöglichkeiten

Damit nicht ständig Adressen eingegeben werden müssen, kann durch Tastendruck in beide Richtungen gescrollt und gepaged werden.

Hierzu eine Erläuterung:

```
        I  O  P    nach oben = rückwärts
        K  L Enter nach unten = vorwärts
        ↓  ↓  ↓
           .
           .
Paging  Scrollen
           .
           .
           .
(ständig eine Zeile)
```

Diese Steuerungsmöglichkeiten und einige andere Optionen werden nach Aufruf des Disassemblers immer auf einer Titelseite angezeigt.
Auf dieser Titelseite muß die Anfangsadresse (hex oder dez), ab der, der Speicher untersucht werden soll, eingegeben werden (Eingabe abschließen mit ENTER). Nach dieser Eingabe erscheint folgendes Menue:

```
D DISASSEMBLER
M MEMORY EDITOR
S CODE STORAGE
```

Durch Drücken der entsprechenden Taste kann man die gewünschte Programm-Routine aufrufen.
Mit diesen Tasten kann dann auch von einer Programm-Routine direkt in eine andere gesprungen werden. Um vom Code-Storage-Mode in den Disassembler zu gelangen, muß man allerdings Symbol Shift + D drücken, da D im Eingabe-Mode als Hexadezimal-Ziffer 13 interpretiert wird.
Weitergehende Informationen für den Dissassembler sind auf dem mitgelieferten Cartridge vorhanden.
Wenn Sie das Cartridge mit RUN starten, erhalten Sie ein Menue, mit dem Sie die Beschreibungen und den Disassembler selbst aufrufen können.

Weiterhin gestattet das Menue, die Einleseprogramme, die für den Benutzer wichtig sind, zu laden.
Um eine schnelle Einlesezeit und eine einfache Korrekturmöglichkeit zu gewährleisten wurde das Einleseprogramm recht aufwendig gestaltet und ausführlich dokumentiert.
Der Datenbereich erfolgt ab Zeile 1000 (in Zehnerschritten) in Data-Statements. Nach Starten des Einleseprogramms muß immer eine Zeilennummer eingegeben werden; und zwar die Zeilennummer, ab der in den Speicher übertragen werden soll (anfangs 1000 eingeben).
Dies hat den Vorteil, daß nach der Korrektur eines Fehlers, die Zeilennummer eingegeben werden kann, somit entfällt die Wartezeit, die normalerweise entsteht, wenn der ganze Speicherbereich ab Zeile 1000 wieder übertragen werden muß.
Ein ganz wichtiger Hinweis:
Nachdem der M-Code korrekt implementiert wurde, muß er erst abgespeichert werden (Save „disas" Code 49152,5123).
Der M-Code darf nicht aufgerufen werden, da er sonst seine Relokationsfähigkeit verliert!
Der so erzeugte Maschinen-Code kann dann in jedem Speicherbereich mit 5123 freien Bytes geladen und mit „Randomize USR (Ladeadresse)" aufgerufen werden.

Noch einige nützliche Pokes:

(hier auch M-Code 49152 laden – Pokes ausführen – abspeichern – jetzt erst aufrufen):

1. Wenn Sie vom Disassembler mittels Q ins Basic zurückgekehrt sind, meldet sich das Basic mit „Ø ok. bzw. Ø Program finished."
   Falls Sie den Disassembler in andere Programme integrieren wollen, kann dies unterdrückt werden, in dem der Return RST 08.DEFBFF überschrieben wird durch Return (201,C9), Änderung: Poke 49670ØØ1.

2. Bewahren der Relokationsfähigkeit auch nach dem Aufruf. Wenn Sie den Disassembler zum Relokieren nicht immer erst neu laden wollen, können Sie eine kleine Verschieberoutine anhängen und folgende Pokes vornehmen:

   49154,Ø
   49155,Ø
   49165,Ø
   49167,Ø

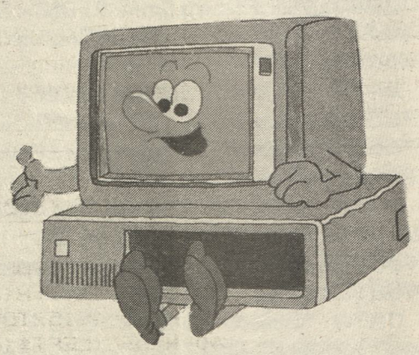
```
100 REM ------------------------
110 REM +SPECTRUM-DISASSEMBLER+
120 REM ------------------------
130 REM
140 REM Code-Einlese-Programm
150 REM
160 REM © Volker Martin (1985)
170 REM   Sebald-Heyden-Str. 39
180 REM   8500 Nuernberg 50
190 REM   0911/483459
200 REM
210 REM ------------------------
220
230
240 REM RAMTOP setzen----------
250
260 CLEAR 49151
270
280 REM Hex-Dez-Umwandlung-----
290
300 DEF FN b(b$)=16*FN d(b$(1))
+FN d(b$(2))
310 DEF FN d(d$)=CODE d$-48-39*
(d$>"9")
320
330 REM Zeilennummer-Eingabe---
340
350 PRINT AT 7,0;"Bitte die Zei
lennummer eingeben,"''"ab der de
r Code in den Speicher"''"uebert
ragen werden soll!"
360 INPUT zeil
370 IF zeil<1000 OR zeil>2010 O
R zeil/10<>INT (zeil/10) THEN LE
T zeil=1000: REM Default-Wert
fuer falsche Eingaben
380
390 REM Startadresse-----------
400
410 LET adr=49152+(zeil-1000)*5
420 RESTORE zeil
430 CLS
440
450 REM POKE-Schleife----------
460
470 FOR i=zeil TO 2010 STEP 10
480  POKE 23692,255
490  PRINT i;" ";
500  READ a$,pruef
510  IF LEN a$<>100 THEN PRINT
"***falsche Stringlaenge!": POKE
23692,1: PAUSE 0: LIST i: STOP
520  LET test=0
530  FOR j=1 TO 99 STEP 2
540   LET h$=a$(j TO j+1)
550   IF h$(1)<"0" OR h$(1)>"f"
OR h$(1)>"9" AND h$(1)<"a" THEN
PRINT "***Zeichen ";j;" = ";h$(
1): POKE 23692,1: PAUSE 0: LIST
i: STOP
560   IF h$(2)<"0" OR h$(2)>"f"
OR h$(2)>"9" AND h$(2)<"a" THEN
PRINT "***Zeichen ";j+1;" = ";h
$(2): POKE 23692,1: PAUSE 0: LIS
T i: STOP
570   LET byte=FN b(h$)
580   POKE adr,byte
590   LET test=test+byte
600   LET adr=adr+1
610  NEXT j
620  IF pruef<>test THEN PRINT
"***Pruefsummenfehler"'test;" -
richtig waere ";pruef;".": POKE
23692,1: PAUSE 0: LIST i: STOP
630  PRINT "OK."
640 NEXT i
650
660 REM CODE abspeichern-------
670
680 CLS
690 PRINT AT 12,3;"M/C korrekt
implementiert."
700 SAVE "DISAS"CODE 49152,5123
: REM bzw. SAVE *"m";1...
710 CLS
720 PRINT AT 12,0;"Zum Verifizi
eren, Band zurueck-"''"spulen un
d starten..."
730 VERIFY "DISAS"CODE : REM
bzw. VERIFY *"m";1...
740 CLS
750 PRINT "Aufnahme ist in Ordn
ung."''''"DISASSEMBLER ist abgesp
eichert"''"als ""DISAS""CODE 491
52,5123."''''"DISAS kann in jeden
 freien Spei-"''"cherbereich mit
 5123 Bytes"''"geladen werden."
760 PRINT ''"Hoechste Ablegeadr
esse: 60245"''"Gegebenenfalls RA
MTOP aendern!"
770 PRINT ''"Aufruf mit RAND US
R Ladeadresse."
780
790 REM Datenbereich-----------
800
1000 DATA "606936c311620019e5eb6
06923732372e1e511710f19eb2100c0a
fed42444de1e5af7eed52e1303a23fec
32815fecd2811fe",5981
1010 DATA "2120eb7efed720e6237ef
ecf20e0180a237efec038d8fed030d4e
57e2b6e67afed42dde1dd7400dd75ffd
de5e12318befdcb",7515
1020 DATA "30defd3631002a6c5ce52
a6e5ce5210020226e5c65226c5c22005
b160000cd4acd2a7b5ce521d7cf019c0
309227b5ccd56cd",4692
1030 DATA "1601049a9894929b999c9
7209d2093959a909a9a9497919694991
60604492e2e2e2e2e2e2e2e2e5365697
465207a75727565",4888
1040 DATA "636b1607044f2e2e2e2e2
e2e5363726f6c6c656e207a757275656
36b160804502e2e2e2e2e2e2e2e2e5a6
5696c65207a7572",3655
1050 DATA "7565636b160a044b2e2e2
e2e2e2e2e536569746520766f7277616
572747316 0b044c2e2e2e2e5363726f6
c6c656e20766f72",3906
1060 DATA "776165727473160c04454
e5445522e2e2e5a65696c6520766f727
76165727473160e04432e2e2e2e2e2e2
e2e447275636b65",3843
1070 DATA "726175736761626516 0f0
4482e2e2e4865782d44657a2d4461727
374656c6c756e67161104512e2e2e2e2
e2e2e4d6f647573",3890
1080 DATA "207665726c617373656e1
6030562792056 6f6c6b6572204d61727
4696e203139383516150545727374652
041647265737365",4123
1090 DATA "3a20ffe1227b5c21005bc
d56cd1615141401ffcd12cecdbececd5
6cd14003c20ffcd56cefe482005cdcbc
e18ddfe512015cd",5983
1100 DATA "4acde1226e5ce1226c5cf
dcb309efdcb31cecffffe0d284ffe303
8bcfe4730b8fe3a380cfe4138b0fdcb3
44620aac609e60f",6686
1110 DATA "fdcb34462008ed6f23ed6
f2b18985fe52a005b011027afed4230f
b09af29e529293808c1c509380319300
326006b22005be1",4487
1120 DATA "e11892cd4acdcd56cd160
8075b445d2020444953415353454d424
c4552160a075b4d5d204d454d4f52592
045444954 4f5216",3936
1130 DATA "0c075b535d2020434f444
52053544f52414745ffcd56cefe4d280
efe532003c3cfc3fe4420eec3f0c5cd4
acd2a005b0e18cd",5388
1140 DATA "64c30d20fa3efddbfecb5
728e7cb4f28dc3efbdbfecb472003c38
2c03efedbfecb5f2003cd72cd3edfdbf
ecb57200c118001",6977
1150 DATA "afed5222005b5718bccb4
7200a3edfdbfecb4728f81804cb4f202
2cd91cdcd56cd160000ffaf1ec8ed522
2005bcd64c31eb8",5821
1160 DATA "19cd56cd1615000d0d0df
f3ebfdbfecb672005cdcbce18bdcb572
00522005b18b4cb47200a3ebfdbfecb4
728f81804cb4f20",5356
1170 DATA "0ee511b800afed5222005
be1cd64c3c3c1c20604fd3652ffcdfcc
dcd56cd2020ffe57efdcb57d6cde0cdf
dcb5796237e23cd",7341
1180 DATA "e0cd10ecfdcb34462805f
13e20d7c9e1cd56cd2020ff06087e23f
ea5301ffe203021e52a7b5ce521d7cfd
5110c0419d1227b",5942
1190 DATA "5c3e90d7e1227b5ce1180
bcbbffdcb57d6d7fdcb579610cec92a0
05b1e0cafed52cd4acd0618fd3652ffc
de2c53e0dd72310",6525
1200 DATA "f75a4acd56cd16000f507
2756566737 56d6d653a24ff2a6c5c3a6
e5cfd7234cda2cefd3534cd56cd16011
a23ffcda2ce326e",5778
1210 DATA "5ccd56cd160c00ff2a005
bcde2c53e3cd77bcde0cd3ef7dbfe2fe
61f20693eefdbfe2fe61f20603efbdbf
ecb472003c382c0",6781
1220 DATA "cb5728513efddbfe2fe60
d20483efedbfecb5f2840cb4f200c4a5
aed536c5ccdb3c5c3ebc33e7fdbfecb6
72828cb572003c3",6568
1230 DATA "b3c23edfdbfe2fe607207
13ebfdbfecb672006cdcbcec3cfc32fe
6072893c356c5188ecd56cefecd2003c
3f0c5fe303880fe",7319
1240 DATA "4730ecfe3a380cfe4138e
4fdcb344620dec609e60f0eff477b580
604fdcb34462819d66430fcc66487478
738098738068038",5525
1250 DATA "03833001af5fc31bc4cb2
3cb23cb23cb23cb231710fb18edcb472
80a3edfdbfecb4728f81804cb4f282fc
db3c5cd91cd2b22",5866
1260 DATA "005b5e2a6c5caf4fed523
002676f226c5c2a005b1e0cafed52cd5
6cd160000ffcde2c55ac3ebc306182b5
ee52a6c5cafed52",5415
1270 DATA "3002676f226c5ce110ed2
```

```
2005bc3cfc342cb47280a3ebfdbfecb4
728f81806cb4f28354373afb1280173c
db3c5c5d5e5cdfe",6150
1280 DATA "0de1d1e55e2a6c5c19226
c5ce1c14a2322005b1e0b19cd56cd161
5000d0dffcde2c558c3ebc306185ee52
a6c5c19226c5ce1",5229
1290 DATA "2310f322005bc3cfc3cd5
6cd16000f20202020202020202020202
02020202016011a202020202020160c0
0ffcde2c5cd56cd",3715
1300 DATA "20202020ffc9cdfccdcd5
6cd2020ff7ecde0cdc921d7cf1114041
90618af57772310fc2a005baffd7776f
d7777cd4acd0e18",6068
1310 DATA "e5e5dde121d7cf114703a
fed5257e5dde5e10600cdfccd3e20d73
ef8fda676fd7776f3c5d5e5061721d7c
f119c0319ebdde5",7501
1320 DATA "e12bc51b1a471b1a4f1ba
fed42c13018c51a471b1a4fafdde5e1a
fed42c1300410dc1804fdcb76de06092
1d7cf11b501195e",5338
1330 DATA "235623e5dde5e1afed52e
12806380810ee1804fdcb76e6e1d1c1f
bfdcb76662003c319c8fd7e77a72877f
d7e76e6e020277e",6956
1340 DATA "f5e6c007073c0f0f0ffdb
676fd7776f1e63f200304237ec650cd5
6cd6578706f3dffcd6dcec9070707c5d
5f5e5210000e5e5",6214
1350 DATA "d1c1e15e3d280c23563d2
807234e3d28022346dd7efff5dd7efee
63f2805f1e63f1801f1c650cdb62acde
32df1d1c14705fd",6126
1360 DATA "35773e1ffda676fd7776c
97efe3e304bf3c5e5470421d7cfd511c
70119d13e79be2330fc10fafbcdd1cde
1c17ea72808fe33",6829
1370 DATA "2804fe3520130423d55e7
b179f5719d1cd56cd20ffcda2cec9fe3
42004fd367701fe38c0fdcb76a6c9fea
03031fe862819fe",6361
1380 DATA "882815fe8c2811cd56cd6
96e76616c696420636f6465ffc9cd56c
d7365726965732dffe60fcd6dcefd777
7c9fec03046fea6",6674
1390 DATA "30d3cd56cd73746b2dffe
6072009cd56cd7a65726fffc93d2008c
d56cd6f6e65ffc93d2009cd56cd68616
c66ffc93d200bcd",6220
1400 DATA "56cd70692f32ffc91899c
d56cd74656effc9fee03012fec63089c
d56cd73746b2d6d656d2dff181bfeef2
007fdcb76a6c328",6754
1410 DATA "c6fee630d1cd56cd67657
42d6d656d2dffe60ff630d7c9fdcb765
e2812fdcb769ecd56cd4445464220ff7
ecd6dcec97efedd",7336
1420 DATA "2006fdcb76c61808fefd2
03afdcb76cedd2323047ee5f521d7cf1
e0819e6f80f0f0f5f19f1f5e607c5470
47e0710fdc13812",5939
1430 DATA "f1e1cd56cd494e4445582
04552524f52ffc9f1e1fecb2003c38cc
bfeed2003c3e8cbcb7f201fcb77281bf
e762817cd56cd4c",6661
1440 DATA "44202020ffcd16cfcd56c
d2cffcbb7cd08cfc9f5cb7f281dcb772
806e607fe062013f1f5e63821d7cfc62
85f19cdd1cdf1cd",6865
1450 DATA "08cfc9f1f5e6c7fe07304
9fe043845fe06281efe052808cd56cd4
94eff1806cd56cd4445ffcd56cd43202
0fff1cd16cfc9f1",6552
1460 DATA "04cd56cd4c44202020ffc
d16cf3e2cd7affdb6762805dd7e02180
3dd7e01cd6dcec9f1f5e6cffe012843f
e032834fe092857",5992
1470 DATA "fe0b2821fec12812fec52
061cd56cd5055534820fff1cda2cfc9c
d56cd504f502020ff18f0cd56cd44454
32020ff18e5cd56",6423
1480 DATA "cd494e432020ff18dacd5
6cd4c44202020fff1cda2cf3e2cd7dd6
602dd6e01cda2ce0404c9cd56cd41444
42020fff1cd7ccf",6343
1490 DATA "cd56cd2cffcda2cfc9f12
3fe222019cd56cd4c4420202028ff040
4cdbececd56cd292cffcd7ccfc9fe2a2
01bcd56cd4c4420",6431
1500 DATA "2020ffcd7ccfcd56cd2c2
8ffcdbece04043e29d7c9fee32010cd5
6cd4558202020285350292cff18c9fee
9200fcd56cd4a50",6048
1510 DATA "20202028ffcd7ccf18d5f
ef9200ecd56cd4c4420202053502cff1
8a4f5e6c7fec42016cd56cd43414c4c2
0ff0404f1cdf5ce",6249
1520 DATA "3e2cd7cdbecec9fec2200
bcd56cd4a50202020ff18e4fec0200ec
d56cd5245542020fff1cdf5cec9f1f5e
6e7fe202029cd56",7104
1530 DATA "cd4a52202020fff1e618c
df5ce3e2cd72304dd7e015ffe8030031
91806ed445fafed52cda2cec9f1fe102
```



```
00bcd56cd444a4e",6092
1540 DATA "5a20ff18d9fe18200bcd5
6cd4a52202020ff18cafec3200fcd56c
d4a50202020ffcdbece0404c9fecd200
bcd56cd43414c4c",5757
1550 DATA "20ff18ebf5e6c7fec7202
ccd56cd5253542020fff1e638cd6dcef
e082004fdcb76defe282010cd56cd2c4
6502d43414c43ff",6561
1560 DATA "fdcb76e6c9f1fed32017c
d56cd4f5554202028ff7ecd6dcecd56c
d292c41ff04c9fedb2015cd56cd494e2
02020412c28ff7e",6420
1570 DATA "cd6dce3e29d704c9fe322
017cd56cd4c4420202028ffcdbececd5
6cd292c41ff0404c9fe3a2014cd56cd4
c44202020412c28",5393
1580 DATA "ffcdbece3e29d718e51e7
421d7cf195f7e23feff2810bb280823c
b7e28fb2318effbcdd1cdc9fbcd56cd4
94e56414c494420",6431
1590 DATA "434f4445ffc9dd2304aff
db6762812dd23dd7e00e60ffe062807f
e0e2803c368c8fdcb76d6dd7e00f521d
7cf1189011915fe",6040
1600 DATA "403005e6380f1808e6c00
f0f0f0fc61c5f19cdd1cd3e20d7f1fe4
0380df5e6380f0f0fc630d73e2cd7f1c
bb7cd08cfc904dd",5665
1610 DATA "23affdb676280405c368c
8dd7e00cb7f2036cb772832cb57202ec
b4f202ccb472015cd56cd494e202020f
fcd16cfcd56cd2c",5511
1620 DATA "284329ffc9cd56cd4f555
420202843292cffcd16cfc91857cb472
83304042323cd56cd4c44202020ffcb5
f2012cd56cd28ff",5178
1630 DATA "cdbececd56cd292cffcda
2cfc9cda2cfcd56cd2c28ffcdbece3e2
9d7c9cb5f2008cd56cd5342ff1806cd5
6cd4144ffcd56cd",7389
1640 DATA "432020484c2cffcda2cfc
91ef5c35fcbd5c501170021d7cf11140
419545d23edb0c1047812d1e13a6e5ca
7280458191811cd",5300
1650 DATA "56cd171400ff3e20d77ec
d6dce2310f6fd3652ffcd56cd0dff0d2
803c30ec63e7fdbfecb572006cdddcec
3b3c23efddbfecb",6893
1660 DATA "4f2006cdddcec3cfc3220
05b3efbdbfecb472006cdddcec382c03
efedbfecb5f2003cd72cd3edfdbfe2fe
6072809cdddcecd",7292
1670 DATA "ddcec3f0c53ebfdbfecb6
72009cdcbcecddcec3f0c5cb572003c
309c6cb4f280ccb47200d3ebfdbfecb4
728f80e01c30ec6",7100
1680 DATA "1887f5c5d5e5cdaf0de1d
1c1f1c922fe5bf5e122fc5be37e23fef
f2803d718f7e32afc5be5f12afe5bc9f
3e5d5c5f5fdcb31",8459
1690 DATA "ce2a005be506c0cdaf0ee
122005bfdcb318ef1c1d1e1c9f3e5d5c
5f5060311ff572edfc562010008edb83
e0884570608c51e",6245
1700 DATA "1f2eff012000edb825c11
0f2c110e213ebaf0e080620772310fc2
46f0d20f5f1c1d1e1fbc9f57ecb7f200
4d72318f7e67fd7",6207
1710 DATA "f1c9f5fdcb34462810fe6
4300c3e20d7f1f5fe0a30033e20d7f1c
d6dcec9d5ed5b005bd522005bcd12ced
1ed53005bd1cda2",6711
1720 DATA "cec9f5c5e5fdcb3446283
72a005b011027afed42302c3e20d72a0
05b01e803afed42301e3e20d72a005b0
16400afed423010",4829
1730 DATA "3e20d72a005b0e0aafed4
230033e20d7e1c1f1c9af32085c3a085
ca728fa3a085cf53a045cfeff20f9f1c
9e5d5c5f50602af",5960
1740 DATA "fdb634280fd1d50e00af4
75fcdb62acde32d1815f1f50f0f0f0ff
5e60ff630fe3a3802c607d7f110edf1c
1d1e1c9e5d5c5f5",6837
1750 DATA "3a6e5ca728054c550018c
c7ccd6dce7dcd6dcef1c1d1e1c9f5e52
37e2b6e67cda2cee1f1c9f53a6e5c2f3
26e5c3ebfdbfecb",7083
1760 DATA "6728f8f1c9e521d7cf111
40419af570618862310fc5fe1ed52220
05bc9f5e5e6380f0f21d7cfc6655f19c
dd1cde1f1c9f5cb",6675
1770 DATA "77280d04dd7e01cd6dcef
1c9f50f0f0fe607e5d55f21d7cf197ef
5d7f1fe28204daffdb676283efe04283
a043e49d73e58fd",6203
1780 DATA "cb764e28013cd7fdcb765
62805dd7eff1803dd7e01f53e20fdcb3
446280ef1f5cb7f3e2b2806f1ed44f53
e2dd7f1cd6dce18",6233
1790 DATA "06cd56cd484cff3e29d7d
1e1f1c9f5affdcb76462808cd56cd495
8ff1814fdcb764e2808cd56cd4959ff1
806cd56cd484cff",6709
1800 DATA "f1c9f5e6302008cd56cd4
```

```
243ff1826fe102008cd56cd4445ff181
afe302013f1cb7f2807cd56cd4146ffc
9cd56cd5350ffc9",6372
1810 DATA "cd7ccff1c942434445484
c28410040004070700e4002020202020
2fd0202020202020202020100000544
000404144442020",2707
1820 DATA "41ac00414443202041ac0
053554220a0000000534243202041ac0
0414e4420a0000000584f5220a000000
04f522020a00000",2910
1830 DATA "0043502020a04edada004
ec3c30050cf50c5d000cd004e4fd0024
c44202020284243292cc107524c43c10
845582020204146",4209
1840 DATA "2c4146a70a4c442020204
12c284243a90f525243c1124c4420202
0284445292cc117524cc11a4c4420202
0412c284445a91f",3325
1850 DATA "5252c1274441c12f4350c
c375343c63f4343c67648414cd4c9524
5d4d94558d8eb455820202044452c48c
cf344c9fb45c9ff",5848
1860 DATA "444e45c745524554ce464
94d20b0474c44202020492cc14d52455
4c94f4c44202020522cc156494d20b15
74c44202020412c",4005
1870 DATA "c95e494d20b25f4c44202
020412cd2675252c46f524cc4a04c44c
9a14350c9a2494ec9a34f5554c9a84c4
4c4a94350c4aa49",5640
1880 DATA "4ec4ab4f5554c4b04c444
9d2b1435049d2b2494e49d2b34f5449d
2b84c4444d2b9435044d2ba494e44d2b
b4f5444d2ff524c",6208
1890 DATA "43a0525243a0524c20a05
25220a0534c41a0535241a0534c41a15
3524ca0424954a0524553a0534554a0e
91da323f22d4d34",4933
1900 DATA "6034b7361c37a1375d388
06a756d702d747275e565786368616e6
7e564656c6574e573756274726163f46
d756c7469706cf9",5768
1910 DATA "6469766973696fee746f2
d706f7765f26ff26e6f2d262d6eef6e6
f2d6c2d6571ec6e6f2d67722d6571ec6
e6f732d6e6571ec",5793
1920 DATA "6e6f2d677274f26e6f2d6
c6573f36e6f732d6571ec61646469746
96fee7374722d262d6eef7374722d6c2
d6571ec7374722d",5682
1930 DATA "67722d6571ec737472732
d6e6571ec7374722d677274f27374722
d6c6573f3737472732d6571ec7374727
32d6164e476616c",5907
1940 DATA "a47573722da4726561642
d69ee6e65676174e5636f64e57661ec6
c65ee7369ee636ff37461ee6173ee616
3f36174ee6cee65",6861
1950 DATA "78f0696ef47371f27367e
e6162f3706565eb69ee7573722d6eef7
37472a4636872a46e6ff46475706c696
36174e56e2d6d6f",6728
1960 DATA "642ded6a756df073746b2
d646174e16465632d6a722d6efa6c657
3732db0677265617465722db0656e642
d63616ce3676574",5679
1970 DATA "2d617267f47472756e636
174e566702d63616c632db2652d746f2
d66f072652d73746163eb13001700250
027002b002f005f",4471
1980 DATA "00650095008d026e04a90
4a109f309110a220af80cff0ca00fa80
f91135415af15c315c615d3152d16331
68f16901616171b",3733
1990 DATA "177a178017481a161b011
c0c1c9625ae259527bc27c5325a336e3
8ff5a3f3f33333f3f33333f3f333e3e3
33f3f3f3f303030",3451
2000 DATA "303f3f3e3f333333333f3
e3f3f303e3e303f3f1e1e0c0c0c0c1e1
e303030303030303f3f333f3f3f333333
33f3f333f3f3030",2525
2010 DATA "303f3f333f3f3e37333f3
f303f3f033f3f3f3f0c0c0c0c0c0c333
333333333333f3f0000003f3f000000000
000181800000000",1752
```

Übrigens:
Große Firmen erreichen „kleine Kunden“ durch ANZEIGEN-WERBUNG!

Datenübertragung auf dem C-64-Computer – leicht gemacht!

Terminalprogramm für Ihren C-64

Die Steuerung
Mit den Cursortasten kann man die gewünschte Funktion anwählen. Mit der Return-Taste kann man die Funktion ausführen. Man geht auf die Funktion Directory und drückt Return, dann erscheint das Directory auf dem Bildschirm.
Die einzelnen Funktionen:

Terminalprogramm
- Dient zur Übertragung von Daten

Besondere Funktionen:
F1 = Empfangene Daten übernehmen (Rahmen : Blau)
F7 = Empfangene Daten nicht übernehmen (Rahmen : Cyan)
F2, F4, F6, F8 = Textspeicher 1, 2, 3, 4 senden
Commodore&1-8=Tastenspeicher 1-8 senden
⇦ = Zurück ins Menü

Speicher löschen
- Der Speicher wird gelöscht

Speicher laden
- Lädt ein File in den Speicher

Besondere Funktionen:
F1 = Pause
F7 = Weiter
⇦ = Abbrechen

Speicher saven
- Speichert den Speicherinhalt ab

Speicher drucken
- Druckt den Speicherinhalt

Speicher auf Bildschirm
- Speicher erscheint auf dem Bildschirm

Besondere Funktionen:
F1 = Pause
F7 = Weiter
⇦ = Abbrechen

SEQ-File auf Bildschirm
- Lädt ein SEQ-File, wird aber nicht in den Speicher übernommen.

Besondere Funktionen:
F1 = Pause
F7 = Weiter
⇦ = Ende

SEQ-File drucken
- Ein SEQ-File wird direkt gedruckt.

SEQ-File kopieren
- Es werden nur bestimmte Teile eines Files in den Speicher geholt.

Besondere Funktionen:
F1 = Pause
F7 = Weiter
F3 = Text übernehmen
F5 = Text nicht übernehmen
⇦ = Abbrechen

Directory
- Das Directory erscheint auf dem Bildschirm.

Besondere Funktionen:
⇦ = Menü

Parameter ändern
- Parameter können verändert werden

Disk Befehl
- Es können Diskettenoperationen wie Scratchen vorgenommen werden.

Textspeicher eingeben
- Man kann einen Text in einem der 4 Textspeicher schreiben. Jeder Textspeicher darf bis zu 500 Zeichen lang sein.

Textspeicher ansehen
- Einen der 4 Textspeicher ansehen

Besondere Funktionen:
F1 = Pause
F7 = Weiter
⇦ = Ende

Textspeicher saven
- Man kann einen der 4 Textspeicher saven

Textspeicher laden
- Man kann einen der 4 Textspeicher laden

Tasten belegen
- Man kann 8 Tasten mit max. 39 Zeichen belegen.

Anmerkung: Wenn man nach der Taste gefragt wird, kommt man mit F7 wieder ins Menü

Tastenbelegung saven
- Die Belegung aller Tasten wird gespeichert

Tastenbelegung laden
- Die Belegung aller Tasten wird geladen

Tastenbelegung auf Bildschirm
- Die Belegung der Tasten erscheint auf dem Bildschirm.

Eingabe der Listings:
1 Das Listing DATA eintippen und starten. MASP wird dabei erzeugt.
2 Das Listing LADER eingeben
3 Das Listing TERMINALPROGRAMM eingeben
Will man das Programm benutzen, so muß man den LADER laden und starten.

Konfiguration:
C64
Floppy Disk
Akustikkoppler
evtl. Drucker

Achtung!
Es kann bei manchen Druckern vorkommen, daß sie nicht auf die Druckroutine reagieren. Abhilfe: Druck-Kanal-Öffnung in den Zeilen 1400, 1630 umändern. Achtung, die logische Filenummer muß bleiben!!!

Erstellen von „MASP“

```
10 DIMH(75):FORI=0TO9
20 H(48+I)=I:H(65+I)=I+10:NEXT
30 FORI=49152TO50378:READA$
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))
50 D=H(H)*16+H(L):S=S+D:POKEI,D
60 A=A+1:IFA<20THENNEXT:A=-1
65 PRINT"ZEILE:"1000+Z;
70 READV:Z=Z+1:IFV=STHENPRINT" OK.":GOTO85
80 PRINT" PRUEFSUMMENFEHLER !";999+Z:STOP
85 IFA<0THEN100
```

```
90 S=0:A=0:PRINT:NEXT
100 OPEN2,8,2,"MASP,P,W"
110 PRINT#2,CHR$(0);CHR$(192);
120 FORI=49152TO50378:PRINT#2,CHR$(PEEK(I));:NEXT
130 CLOSE2
1000 DATAA9,00,85,FE,A9,40,85,FF,A2,02,20,C6,FF,A0,00,20,CF,FF,20,D2, 2722
1001 DATAFF,91,FE,E6,FE,D0,02,E6,FF,4C,3A,C0,24,90,50,EB,A5,FE,8D,F0, 3454
1002 DATA03,A5,FF,8D,F1,03,20,CC,FF,A9,02,20,C3,FF,60,EA,EA,EA,A5,CB, 3118
1003 DATAC9,04,D0,09,A5,CB,C9,03,D0,FA,4C,20,C0,A5,CB,C9,39,D0,D1,4C, 2871
1004 DATA24,C0,EA,EA,EA,A2,02,20,C6,FF,20,CF,FF,20,D2,FF,4C,70,C0,24, 2986
1005 DATA90,50,F3,20,CC,FF,A9,02,20,C3,FF,60,A5,CB,C9,04,D0,09,A5,CB, 2865
1006 DATAC9,03,D0,FA,4C,63,C0,A5,CB,C9,39,D0,DE,4C,67,C0,EA,EA,A9,00, 3093
1007 DATA85,FE,A9,40,85,FF,A9,36,85,01,A2,02,20,C9,FF,A0,00,B1,FE,20, 2640
1008 DATAD2,FF,E6,FE,D0,02,E6,FF,A5,FE,CD,F0,03,D0,EE,A5,FF,CD,F1,03, 3826
1009 DATAD0,E7,A9,02,20,C3,FF,4C,85,C1,EA,EA,A9,00,85,FE,A9,40,85,FF, 3139
1010 DATAA9,36,85,01,A2,04,20,C9,FF,A0,00,B1,FE,20,D2,FF,E6,FE,D0,02, 2793
1011 DATAE6,FF,A5,CB,C9,39,D0,03,4C,F5,C0,A5,FE,CD,F0,03,D0,E5,A5,FF, 3559
1012 DATACD,F1,03,D0,DE,A9,37,85,01,20,CC,FF,A9,04,20,C3,FF,60,EA,EA, 2947
1013 DATAEA,A2,02,20,C6,FF,20,CF,FF,8D,FA,03,A2,04,20,C9,FF,AD,FA,03, 2851
1014 DATA20,D2,FF,4C,30,C1,24,90,50,E3,20,CC,FF,A9,02,20,C3,FF,A9,04, 2618
1015 DATA20,C3,FF,60,A5,CB,C9,39,D0,E8,4C,22,C1,EA,EA,EA,EA,EA,EA,EA, 3585
1016 DATAA0,00,84,FE,A9,40,85,FF,A9,36,85,01,A5,FE,CD,F0,03,D0,0C,A5, 2776
1017 DATAFF,CD,F1,03,D0,05,A9,37,85,01,60,B1,FE,20,D2,FF,E6,FE,D0,02, 2993
1018 DATAE6,FF,A5,CB,C9,04,D0,09,A5,CB,C9,03,D0,FA,4C,4C,C1,A5,CB,C9, 3219
1019 DATA39,D0,CD,4C,5A,C1,EA,EA,EA,20,CC,FF,A9,37,85,01,60,EA,EA,EA, 3178
1020 DATAA2,02,20,C6,FF,A0,00,84,FE,A9,40,85,FF,20,CF,FF,8D,F6,03,20, 2732
1021 DATAD2,FF,A5,FD,C9,00,D0,0B,AD,F6,03,91,FE,E6,FE,D0,02,E6,FF,4C, 3379
1022 DATAD1,C1,24,90,50,DF,A5,FE,8D,F0,03,A5,FF,8D,F1,03,20,CC,FF,A9, 3153
1023 DATA02,20,C3,FF,60,A5,CB,C9,04,D0,09,A5,CB,C9,03,D0,FA,4C,BA,C1, 2855
1024 DATAA5,CB,C9,39,D0,03,4C,BE,C1,A5,CB,C9,05,D0,0C,A9,00,85,FD,A9, 2814
1025 DATA06,8D,20,D0,4C,BA,C1,A5,CB,C9,06,D0,09,A9,01,85,FD,A9,03,8D, 2503
1026 DATA20,D0,4C,BA,C1,EA,EA,EA,AD,F0,03,8D,F4,03,AD,F1,03,8D,F5,03, 3007
1027 DATAAD,F2,03,85,FE,AD,F3,03,85,FF,20,08,C0,AD,F0,03,8D,F2,03,AD, 2819
1028 DATAF1,03,8D,F3,03,4C,76,C2,EA,EA,AD,F4,03,85,FE,AD,F5,03,85,FF, 3103
1029 DATAAD,F0,03,8D,F4,03,AD,F1,03,8D,F5,03,AD,F2,03,8D,F0,03,AD,F3, 2828
1030 DATA03,8D,F1,03,20,96,C0,AD,F4,03,8D,F0,03,AD,F5,03,8D,F1,03,A5, 2537
1031 DATAFE,8D,F2,03,A5,FF,8D,F3,03,60,AD,F4,03,8D,F0,03,AD,F5,03,8D, 2909
1032 DATAF1,03,60,EA,EA,AD,F2,03,85,FE,AD,F3,03,85,FF,A2,00,A0,00,B1, 2919
1033 DATAFE,C9,00,D0,01,60,20,D2,FF,E8,E0,28,F0,F7,E6,FE,D0,ED,E6,FF, 3654
1034 DATA4C,93,C2,4C,42,C3,EA,EA,20,E4,FF,C9,00,F0,F4,C9,5F,F0,3B,C9, 3218
1035 DATA85,F0,38,C9,88,F0,37,C9,89,F0,36,C9,8A,F0,35,C9,8B,F0,34,C9, 3158
1036 DATA8C,F0,33,C9,81,F0,32,C9,95,F0,31,C9,96,F0,30,C9,97,F0,2F,C9, 3169
1037 DATA98,F0,2E,C9,99,F0,2D,C9,9A,F0,2C,C9,9B,F0,2B,4C,21,C3,60,4C, 2831
1038 DATAA0,C3,4C,90,C3,4C,70,C4,4C,7B,C4,4C,86,C4,4C,91,C4,4C,B0,C3, 2819
1039 DATA4C,BB,C3,4C,C6,C3,4C,D1,C3,4C,DC,C3,4C,E7,C3,4C,F2,C3,4C,FD, 3242
1040 DATAC3,C9,41,90,15,C9,60,B0,03,4C,33,C3,C9,C0,B0,08,4C,3A,C3,69, 2435
1041 DATA20,4C,3A,C3,E9,80,85,9E,20,4A,C3,EA,EA,EA,20,50,C3,4C,B0,C2, 2769
1042 DATA85,9E,20,08,F2,60,EA,EA,20,86,F0,C9,00,F0,33,C9,41,90,0E,C9, 2660
1043 DATA60,B0,08,4C,62,C3,69,80,4C,69,C3,E9,20,85,FD,20,30,C4,A5,FC, 2602
1044 DATAC9,00,D0,16,A0,00,A5,FD,91,FE,E6,FE,D0,0C,E6,FF,A5,FF,C9,C0, 3410
1045 DATAD0,04,A9,40,85,FF,60,EA,EA,EA,EA,EA,A9,01,85,FC,A9,03,8D,20, 2999
1046 DATAD0,4C,B0,C2,EA,EA,EA,EA,A9,00,85,FC,A9,06,8D,20,D0,4C,B0,C2, 3146
1047 DATAEA,EA,EA,EA,A9,6F,A0,CE,85,4E,84,4F,4C,10,C4,A9,97,A0,CE,85, 3111
1048 DATA4E,84,4F,4C,10,C4,A9,BF,A0,CE,85,4E,84,4F,4C,10,C4,A9,E7,A0, 2573
1049 DATACE,85,4E,84,4F,4C,10,C4,A9,0F,A0,CF,85,4E,84,4F,4C,10,C4,A9, 2346
1050 DATA37,A0,CF,85,4E,84,4F,4C,10,C4,A9,5F,A0,CF,85,4E,84,4F,4C,10, 2277
1051 DATAC4,A9,87,A0,CF,85,4E,84,4F,4C,10,C4,EA,EA,EA,EA,EA,EA,EA,EA, 3449
1052 DATAA2,00,A0,00,B1,4E,C9,00,F0,11,20,50,C4,20,50,C3,E6,4E,D0,02, 2168
1053 DATAE6,4F,E8,E0,27,D0,E7,4C,B0,C2,EA,EA,C9,13,F0,13,C9,0D,F0,10, 3106
```



```
1054 DATAA5,FD,20,D2,FF,A9,A4,20,D2,FF,A9,9D,20,B5,C4,60,A9,20,20,D2, 3019
1055 DATAFF,4C,38,C4,C9,41,90,15,C9,60,B0,03,4C,62,C4,C9,C0,B0,08,4C, 2513
1056 DATA69,C4,69,20,4C,69,C4,E9,80,4C,48,C3,EA,EA,EA,EA,A9,C5,A0,00, 2981
1057 DATA85,4F,84,4E,4C,9A,C4,A9,C7,A0,00,85,4F,84,4E,4C,9A,C4,A9,C9, 2594
1058 DATAA0,00,85,4F,84,4E,4C,9A,C4,A9,CB,A0,00,85,4F,84,4E,EA,A0,00, 2356
1059 DATAB1,4E,C9,00,F0,0F,20,50,C4,20,50,C3,E6,4E,D0,EE,E6,4F,4C,9A, 2619
1060 DATAC4,4C,B0,C2,EA,A2,00,86,D4,86,D8,20,D2,FF,A9,90,4C,D2,FF,88, 3221
1061 DATA88,16,88,EC,88,66,88, 904
```

Vorprogramm

```
0 REM ********************************
1 REM *                              *
2 REM *      WRITTEN IN 1985 BY      *
3 REM *                              *
4 REM *  CHRISTOPH SCHAEPERMEIER-HO  *
5 REM *                              *
6 REM *      WELLBRAUCKSWEG 115      *
7 REM *   4250 BOTTROP-KIRCHHELLEN   *
8 REM *                              *
9 REM ********************************
10 POKE53280,14:POKE53281,14
20 PRINT"■EHN    ■LEASE WAIT WHILE LOADING"
30 A$=CHR$(34):POKE56,63
40 PRINT"■X■■■LOAD"A$"MASP"A$",8,1"
45 PRINT"■■■■NEW"
50 PRINT"■■LOAD"A$"BASIC"A$",8"
60 PRINT"■■■■RUN"
70 PRINT"■"
80 FORI=0TO3:POKE631+I,13:NEXT:POKE198,7
```

Hauptprogramm

```
10 CLR:POKE56,64:M$="├──────────────────────────────────────────────────────────────┤"
30 T(1)=52847:T(2)=52887:T(3)=52927:T(4)=52967:T(5)=53007:YR=4:XR=1:P=1
40 T(6)=53047:T(7)=53087:T(8)=53127
41 IFPEEK(1022)<>253THEN70
50 S(1)=PEEK(910)+256*PEEK(911):S(3)=PEEK(914)+256*PEEK(915)
51 S(2)=PEEK(912)+256*PEEK(913):S(4)=PEEK(916)+256*PEEK(917):GOTO100
70 POKE900,38:POKE901,0:POKE902,0:POKE903,0
71 POKE910,0:POKE911,197:POKE912,0:POKE913,199:POKE914,0:POKE915,201
72 POKE916,0:POKE917,203
75 POKE904,0:POKE905,0:POKE906,0:POKE907,0:POKE1022,253
80 S(1)=PEEK(910)+256*PEEK(911):S(3)=PEEK(914)+256*PEEK(915)
81 S(2)=PEEK(912)+256*PEEK(913):S(4)=PEEK(916)+256*PEEK(917)
84 FORI=1TO4:POKES(I),0:NEXT:FORI=1TO8:POKET(I),0:NEXT
94 POKE900,38:POKE901,0:POKE1008,0:POKE1009,64:GOTO100
100 POKE53280,3:POKE53281,3:FS=49152-(PEEK(1008)+256*PEEK(1009))
102 BS=(PEEK(1008)+256*PEEK(1009))-16384:YR=4:XR=1:P=1
110 PRINT"■■■";CHR$(142);
120 PRINT"┌──────────────────────────────────────────────────────────────┐";
130 PRINT"│■FREIER SPEICHER :"FS;TAB(79)"■│";
140 PRINT"│■BENUTZTER SPEICHER :"BS;TAB(39)"■│";
150 PRINT"├──────────────────────────┬──────────────────────────┤";
160 PRINT"│● ■TERMINAL PROGRAMM■│●■ PARAMETER CHANGE■│";
170 PRINTM$;
180 PRINT"│●■ SPEICHER LOESCHEN■│●■ DISK BEFEHL ■     │";
190 PRINTM$;
200 PRINT"│●■ SPEICHER LADEN■    │●■ TEXTSP. EINGEBEN■│";
210 PRINTM$;
220 PRINT"│●■ SPEICHER SAVEN■    │●■ TEXTSP. ANSEHEN■  │";
230 PRINTM$;
240 PRINT"│●■ SPEICHER DRUCKEN■  │●■ TEXTSP. SAVEN■    │";
250 PRINTM$;
260 PRINT"│●■ SPEICH.AUF SCREEN■│●■ TEXTSP. LADEN ■   │";
270 PRINTM$;
280 PRINT"│●■ SEQ-FILE AUF SCR.■│●■ TASTEN BELEGEN■   │";
290 PRINTM$;
```

```
300 PRINT"|●■ SEQ-FILE DRUCKEN■  |●■ TASTENB. SAVEN ■  |";
310 PRINTM$;
320 PRINT"|●■ SEQ-FILE KOPIEREN■|●■ TASTENB. LADEN■   |";
330 PRINTM$;
340 PRINT"|●■ DIRECTORY          ■|●■ TASTENB.AUF SCR.■|";
350 PRINT"└──────────────────────┴──────────────────────┘";
390 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
400 GETA$:IFA$=""THEN400
410 IFA$="■"THEN500
420 IFA$="■"THEN600
430 IFA$="■"THEN700
440 IFA$="■"THEN800
450 IFA$=CHR$(13)THEN900
460 GOTO400
500 P=P+1
510 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
515 IFP>2OTHENP=1:YR=4:XR=1:GOTO540
520 IFP/2<>INT(P/2)THENYR=YR+2:XR=1:GOTO540
530 XR=21
540 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
550 GOTO400
600 P=P-1
610 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
615 IFP<1THENP=20:YR=22:XR=21:GOTO640
620 IFP/2<>INT(P/2)THENXR=1:GOTO640
630 XR=21:YR=YR-2
640 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
650 GOTO400
700 P=P+2
710 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
715 IFP>20THEN760
720 IFP/2<>INT(P/2)THENYR=YR+2:XR=1:GOTO740
730 XR=21:YR=YR+2
740 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
750 GOTO400
760 IFP/2<>INT(P/2)THENYR=4:XR=21:P=2:GOTO740
770 P=1:XR=1:YR=4:GOTO740
800 P=P-2
810 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
815 IFP<1THEN860
820 IFP/2<>INT(P/2)THENXR=1:YR=YR-2:GOTO840
830 XR=21:YR=YR-2
840 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
850 GOTO400
860 IFP/2<>INT(P/2)THENYR=22:XR=21:P=20:GOTO740
870 P=19:XR=1:YR=22:GOTO840
900 ONPGOTO6500,3000,915,2000,1200,4100,1300,940,1400,950,1500,960,4900,5200
901 ONP-14GOTO1600,5800,4000,5900,6000,996
915 POKE1008,0:POKE1009,64:GOTO100
```

```
940 KJ=1:GOTO4100
950 KJ=2:GOTO4100
960 KJ=3:GOTO4100
996 KJ=1:GOTO5200
1000 D$=""
1010 PRINT"■/AME : ■■"
1020 GETA$:IFA$=""THEN1020
1030 IFA$="■"THEN1100
1040 IFA$=CHR$(13)THEN1150
1050 IFASC(A$)<32ORA$="■"THEN1020
1060 D$=D$+A$
1070 PRINT"■/AME : "D$"■ ■■"
1080 GOTO1020
1100 IFLEN(D$)<1THEN1020
1110 D$=LEFT$(D$,LEN(D$)-1)
1120 GOTO1070
1150 IFLEN(D$)<1THEN1020
1160 RETURN
```

```
1200 PRINT"■■■":GOSUB1000:OPEN2,8,2,D$+",S,R":SYS49152:GOTO100
1300 PRINT"■■■":GOSUB1000:OPEN2,8,2,D$+",S,W":SYS49290:GOTO100
1400 OPEN4,4,0:SYS49344:GOTO100
1500 PRINT"■■■":SYS49472:POKE198,0:WAIT198,1:GETA$:GOTO100
1600 PRINT"■■■":GOSUB1000:OPEN2,8,2,D$+",S,R":OPEN4,4,0:SYS49413:GOTO100
2000 PRINT"■■";CHR$(142);
2010 PRINT"■■■■● VALIDATE"
2020 PRINT"■■● LOESCHEN"
2030 PRINT"■■● UMBENENNEN"
2040 PRINT"■■● FORMATIEREN"
```



```
2050 PRINT"■■● FORMATIEREN OHNE ID"
2055 PRINT"■■● MENUE"
2060 YR=3:XR=0:POKE211,XR:POKE214,YR:SYS58732:PRINT"■●":P=1
2100 GETA$:IFA$=""THEN2100
2110 IFA$="■"THEN2200
2120 IFA$="■"THEN2300
2130 IFA$=CHR$(13)THEN2400
2140 GOTO2100
2200 P=P+1:IFP>6THENP=1:GOTO2260
2210 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
2220 YR=YR+3
2230 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
2240 GOTO2100
2260 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●":YR=3:GOTO2230
2300 P=P-1:IFP<1THENP=6:GOTO2360
2310 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
2320 YR=YR-3
2330 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
2340 GOTO2100
2360 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●":YR=18:GOTO2230
2400 ONPGOTO2500,2600,2700,2800,2900,100
2500 PRINT"■■":OPEN1,8,15:PRINT#1,"V":INPUT#1,A,A$,T,S
2510 INPUT#1,A,A$,T,S:PRINTA","A$","T","S:CLOSE1:POKE198,0:WAIT198,1:GETA$
2520 GOTO2000
2600 PRINT"■■■":GOSUB1000:PRINT"■"CHR$(142):OPEN1,8,15:PRINT#1,"S:"+D$:GOTO2510
2700 PRINT"■":INPUT"■■ALTER NAME ";A$:INPUT"NEUER NAME ";N$:OPEN1,8,15
2710 PRINT#1,"R:"+N$+"="+A$:GOTO2510
2800 PRINT"■■■":GOSUB1000:INPUT"■■ID ";ID$:OPEN1,8,15:PRINT#1,"N:"+D$+","+ID$
2810 GOTO2510
2900 PRINT"■■■":GOSUB1000:OPEN1,8,15:PRINT#1,"N:"+D$:GOTO2510
3000 PRINT"■■"CHR$(142)
3010 PRINT"■■■■● BAUDRATE"
3020 PRINT"■■● DATENBITS"
3030 PRINT"■■● STOP-BITS"
3040 PRINT"■■● DUPLEX"
3050 PRINT"■■● PARITY"
3060 PRINT"■■● MENUE"
3070 XR=0:YR=4:POKE211,XR:POKE214,YR:SYS58732:PRINT"■●":P=1
3100 GETA$:IFA$=""THEN3100
3110 IFA$="■"THEN3200
3120 IFA$="■"THEN3300
3130 IFA$=CHR$(13)THEN3400
3140 GOTO3100
3200 P=P+1
3210 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3215 IFP>6THEN3260
3220 YR=YR+3
3230 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3240 GOTO3100
3260 P=1:YR=4:POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3270 GOTO3230
3300 P=P-1
3310 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3315 IFP<1THEN3360
3320 YR=YR-3
3330 POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3340 GOTO3100
3360 P=6:YR=19:POKE211,XR:POKE214,YR:SYS58732:PRINT"■●"
3370 GOTO3330
3400 ONPGOTO3500,3600,3700,3800,3900,100
```

```
3500 PRINT"■BAUDRATE : 300":P=2:C=6
3510 GETA$:IFA$=""THEN3510
3511 IFA$="■"THEN3520
3512 IFA$="■"THEN3540
3513 IFA$=CHR$(13)THEN3580
3514 GOTO3510
3520 P=P+1:IFP>4THENP=1
3530 GOTO3550
3540 P=P-1:IFP<1THENP=4
3550 ONPGOTO3560,3564,3568,3570
3560 PRINT"■■■■■■■■■■■■150 "
3561 C=5:GOTO3510
3564 PRINT"■■■■■■■■■■■■300 "
3565 C=6:GOTO3510
3568 PRINT"■■■■■■■■■■■■600 "
3569 C=7:GOTO3510
3570 PRINT"■■■■■■■■■■■■1200"
3571 C=8:GOTO3510
3580 J=PEEK(900)AND240:J=J+C:POKE900,J
3590 GOTO3000
3600 PRINT"■DATENBITS : 7":P=1:C=32
3610 GETA$:IFA$=""THEN3610
3620 IFA$="■"THEN3660
3630 IFA$="■"THEN3690
3640 IFA$=CHR$(13)THEN3697
3650 GOTO3610
3660 P=P+1:IFP>2THENP=1
3670 IFP=1THENPRINT"■DATENBITS : 7":C=32
3680 IFP=2THENPRINT"■DATENBITS : 8":C=0
3685 GOTO3610
3690 P=P-1:IFP<1THENP=2
3695 GOTO3670
3697 J=PEEK(900)AND159:J=J+C:POKE900,J
3698 GOTO3000
```

```
3700 PRINT"■STOP-BITS : 1":P=1:C=0
3710 GETA$:IFA$=""THEN3710
3720 IFA$="■"THEN3760
3730 IFA$="■"THEN3790
3740 IFA$=CHR$(13)THEN3795
3750 GOTO3710
3760 P=P+1:IFP>2THENP=1
3770 IFP=1THENPRINT"■STOP-BITS :1":C=0
3780 IFP=2THENPRINT"■STOP-BITS :2":C=128
3785 GOTO3710
3790 P=P-1:IFP<1THENP=2
3791 GOTO3770
3795 J=PEEK(900)AND127:J=J+C:POKE900,
     J:GOTO3000
3800 PRINT"■DUPLEX : VOLL":P=1:C=0
3810 GETA$:IFA$=""THEN3810
3820 IFA$="■"THEN3860
3830 IFA$="■"THEN3890
3840 IFA$=CHR$(13)THEN3895
3850 GOTO3810
3860 P=P+1:IFP>2THENP=1
3870 IFP=1THENPRINT"■DUPLEX : VOLL":C=0
3880 IFP=2THENPRINT"■DUPLEX : HALB":C=16
3885 GOTO3810
3890 P=P-1:IFP<1THENP=2
```

```
3891 GOTO3870
3895 J=PEEK(901)AND239:J=J+C:POKE901,J:GOTO3000
3900 PRINT"■PARITY : NO PARITY":P=1:C=32
3910 GETA$:IFA$=""THEN3910
3920 IFA$="■"THEN3960
3930 IFA$="■"THEN3970
3940 IFA$=CHR$(13)THEN3990
3950 GOTO3910
3960 P=P+1:IFP>5THENP=1
3961 GOTO3980
3970 P=P-1:IFP<1THENP=5
3980 IFP=1THENPRINT"■PARITY : NO PARITY  ":C=0
3981 IFP=2THENPRINT"■PARITY : ODD PARITY ":C=32
3982 IFP=3THENPRINT"■PARITY : EVEN PARITY":C=96
3983 IFP=4THENPRINT"■PARITY : MARK       ":C=160
3984 IFP=5THENPRINT"■PARITY : SPACE      ":C=224
3985 GOTO3910
3990 J=PEEK(901)AND31:J=J+C:POKE901,J:GOTO3000
4000 PRINT"■■■■":GOSUB1000:OPEN2,8,2,D$+",S,R":SYS49552
4010 POKE198,0:WAIT198,1:GETA$:GOTO100
4100 PRINT"■■■■          oELCHER | EXTSPEICHER ?"
4120 PRINT"■■■■      1        2        3        4"
4130 XR=5:YR=4:P=1:POKE211,XR:POKE214,YR:SYS58732:B$=STR$(P):PRINT"■"B$
4140 GETA$:IFA$=""THEN4140
4150 IFA$="■"THEN4200
```



```
4160 IFA$="■"THEN4300
4170 IFA$=CHR$(13)THEN4400
4180 GOTO4140
4200 P=P+1
4210 POKE211,XR:POKE214,YR:SYS58732:PRINT"■"B$
4220 IFP>4THEN4270
4230 XR=XR+9
4240 POKE211,XR:POKE214,YR:SYS58732:B$=STR$(P):PRINT"■"B$
4250 GOTO4140
4270 P=1:POKE211,XR:POKE214,YR:SYS58732:PRINT"■"B$
4280 XR=5:POKE211,XR:POKE214,YR:SYS58732:B$=STR$(P):PRINT"■"B$:GOTO4140
4300 P=P-1
4310 POKE211,XR:POKE214,YR:SYS58732:PRINT"■"B$
4320 IFP<1THEN4370
4330 XR=XR-9
4340 POKE211,XR:POKE214,YR:SYS58732:B$=STR$(P):PRINT"■"B$
4350 GOTO4140
4370 P=4:POKE211,XR:POKE214,YR:SYS58732:PRINT"■"B$
4380 XR=32:POKE211,XR:POKE214,YR:SYS58732:B$=STR$(P):PRINT"■"B$:GOTO4140
4400 IFP=1THENI=50432
4410 IFP=2THENI=50944
4420 IFP=3THENI=51456
4430 IFP=4THENI=51968
4431 IFKJ=1THENKJ=0:GOTO4700
4432 IFKJ=2THENKJ=0:GOTO5000
4433 IFKJ=3THENKJ=0:GOTO5100
4440 O=0:PRINT"■■■ EXTENDE = ■-7"
4450 PRINT"┌.┘.":PRINT"■■ ■■";:O=0
4500 GETA$:IFA$=""THEN4500
4510 IFA$=CHR$(13)THEN4650
4515 IFA$="■"THEN4670
4520 IFA$="■"THEN4600
4530 IFA$="■"ORASC(A$)<32THEN4500
4540 POKEI,ASC(A$)
4550 PRINTA$"_ ■■";
4560 I=I+1:O=O+1
4570 IFO=511THEN4500
4580 GOTO4500
4600 IFO=0THEN4500
4610 I=I-1:O=O-1:PRINT"■_ ■■";
4620 GOTO4500
4650 PRINT" ";:GOTO4540
4670 POKEI,0:I=I+1:S(P)=I:X=INT(I/256):Y=I-(X*256)
4680 POKE908+(P*2),Y:POKE909+(2*P),X:GOTO100
4700 PRINT"■■■";
4710 GETA$
4730 IFA$="■"THEN4800
4740 IFA$="_"THEN4850
4780 X=PEEK(I):IFI=S(P)THEN4850
4790 PRINTCHR$(X);:I=I+1:GOTO4710
4800 GETA$:IFA$<>"■"THEN4800
4810 GOTO4740
4850 POKE198,0:WAIT198,1:GETA$:GOTO100
4900 PRINT"■■■"
4910 GOSUB1000
4920 OPEN2,8,2,D$+",S,R"
4930 SYS49237
4940 POKE198,0:WAIT198,1:GETA$:GOTO100
5000 Y=INT(I/256):X=I-(Y*256):POKE1012,X:POKE1013,Y
5005 Y=INT(S(P)/256):X=S(P)-(Y*256):POKE1010,X:POKE1011,Y
```

```
5010 PRINT"■■■":GOSUB1000:OPEN2,8,2,D$+",S,W":SYS49722:GOTO100
5100 Y=INT(I/256):X=I-(Y*256):POKE1010,X:POKE1011,Y
5110 PRINT"■■■":GOSUB1000:OPEN2,8,2,D$+",S,R":SYS49680
5140 S(P)=PEEK(1010)+256*PEEK(1011)
5150 POKE198,0:WAIT198,1:GOTO100
5200 PRINT"■■■"
5210 PRINT"■■■= 1":I=T(1):GOSUB5350
5220 PRINT"-= 2":I=T(2):GOSUB5350
5230 PRINT"-= 3":I=T(3):GOSUB5350
5240 PRINT"-= 4":I=T(4):GOSUB5350
5250 PRINT"-= 5":I=T(5):GOSUB5350
5260 PRINT"-= 6":I=T(6):GOSUB5350
5270 PRINT"-= 7":I=T(7):GOSUB5350
5280 PRINT"-= 8":I=T(8):GOSUB5350
5310 GOTO5400
5350 Y=INT(I/256):X=I-(Y*256):POKE1010,X:POKE1011,Y:SYS49797:PRINT:RETURN
5400 IFKJ=0THEN5500
5410 PRINT"■ ASTE DRUECKEN !!!":KJ=0
5420 POKE198,0:WAIT198,1:GETA$:GOTO100
5500 PRINT"■OELCHE | ASTE SOLL BELEGT WERDEN ?"
5520 GETA$:IFA$=""THEN5520
5530 IFA$="■"THEN100
5540 IFA$="1"THENP=1:GOTO5600
5545 IFA$="2"THENP=2:GOTO5600
5550 IFA$="3"THENP=3:GOTO5600
5555 IFA$="4"THENP=4:GOTO5600
5560 IFA$="5"THENP=5:GOTO5600
5565 IFA$="6"THENP=6:GOTO5600
5570 IFA$="7"THENP=7:GOTO5600
5575 IFA$="8"THENP=8:GOTO5600
5590 GOTO5520
5600 I=T(P):D$="":D=0
5610 PRINT"■";
5620 GETA$:IFA$=""THEN5620
5640 IFA$="■"THEN5700
5650 IFA$=CHR$(13)THEN5750
5660 IFA$="■"ORASC(A$)<32THEN5620
5665 D=D+1:IFD>39THEND=D-1:GOTO5620
5670 D$=D$+A$
5680 PRINTA$"_ ■■";
5690 GOTO5620
5700 IFD$=""THEN5620
5710 D=D-1:D$=LEFT$(D$,LEN(D$)-1):PRINT"■_ ■■";
5720 GOTO5620
5750 FORI=0TOD-1
5760 POKEI+T(P),ASC(MID$(D$,I+1,1))
5770 NEXT
5780 POKET(P)+D,0
5790 GOTO5200
5800 PRINT"■■■":GOSUB1000
5820 POKE1010,255:POKE1011,207:POKE1012,111:POKE1013,206
5830 OPEN2,8,2,D$+",S,W":SYS49722:GOTO100
5900 PRINT"■■■":GOSUB1000:POKE1010,111:POKE1011,206
5910 OPEN2,8,2,D$+",S,R":SYS49680:GOTO100
6000 PRINT"■";:OPEN1,8,0,"$0"
6020 GET#1,A$,B$
6030 GET#1,A$,B$
6040 GET#1,A$,B$
6050 C=0
6060 IFA$<>""THENC=ASC(A$)
6070 IFB$<>""THENC=C+ASC(B$)*256
6080 PRINTMID$(STR$(C),2);TAB(5);
6090 GET#1,B$:IFST<>0THEN6190
6100 IFB$<>CHR$(34)THEN6090
6110 GET#1,B$:IFB$<>CHR$(34)THENPRINTB$;:GOTO6110
6120 GET#1,B$:IFB$=CHR$(32)THEN6120
6130 PRINTTAB(22);:C$=""
6140 C$=C$+B$:GET#1,B$:IFB$<>""THEN6140
6150 PRINTLEFT$(C$,3);:IFLEFT$(C$,3)="SEQ"THENPRINT"  <-";
6160 GETT$:PRINT
6170 IFT$=" "THENPOKE198,0:WAIT198,1:GETT$
6180 IFT$=" "THEN6195
6185 IFST=0THEN6030
6190 PRINT"BLOCKS FREE."
6195 CLOSE1:POKE198,0:WAIT198,1:GETA$:GOTO100
6500 PRINT"■■■ ERMINAL PROGRAMM":PRINT"■■ ■■"
6510 OPEN2,2,0,CHR$(PEEK(900))+CHR$(PEEK(901))
6520 POKE254,PEEK(1008):POKE255,PEEK(1009):GET#2,A$:SYS50064
6530 GET#2,A$
6540 POKE1008,PEEK(254):POKE1009,PEEK(255):CLOSE2:RUN
```


Datenübertragung zwischen APPLE II und COMMODORE 64

Auf dem Gebiet der Personalcomputer gehört der Apple II mit seinen vielen kompatiblen Verwandten zu den Marktgrößen. Der C 64 ist der unbestrittene Marktführer bei den Homecomputern. Deshalb ist es durchaus naheliegend, ein Interface zu konstruieren, welches den Datenaustausch zwischen den beiden Rechnern ermöglicht.
Eine Möglichkeit bieten natürlich fertige, genormte Schnittstellen (z. B. RS-232). Allerdings sind die Hardwarekosten dabei relativ hoch.
Mit diesem Beitrag möchten wir ein einfaches und vor allem sehr preiswertes Verfahren zur Datenübertragung zwischen Apple und Commodore vorstellen.

Zur Kostensenkung verwenden wir Ein- und Ausgänge, die schon in der Grundausstattung der Rechner vorhander sind.
Beim Apple bietet der Game-I/O-Port (Anschluß für Joystick/Paddle) die Möglichkeit, Daten ein- oder auszugeben: Es sind drei Einbit-Eingänge (Pushbuttons) und vier Einbit-Ausgänge (Annunciators) vorhanden. Mit diesen Ein/Ausgabe-Möglichkeiten des Apple-Game-Ports ist leider nur eine serielle Datenübertragung möglich.
Nähere Informationen zur Game-Port-Belegung können Sie dem Apple II Reference Manual entnehmen (Seite 100). Der Game-Port befindet sich in der hinteren, rechten Ecke der Hauptplatine.
Der Commodore hat mit dem Userport ebenfalls eine Ein/Ausgabe-Schnittstelle zur Verfügung. Diese finden Sie an der Rückseite des Computers.
Trotz der Standard-Schnittstellen errreicht unser Interface eine recht beachtliche Übertragungsrate. Im Erprobungsstadium haben wir bereits Raten von ca. 9000 Baud (Bit/Sekunde) gemessen.
Zur Übertragung werden je eine Daten- und ein Strobeleitung (Signalleitung) vom Apple zum Commodore und umgekehrt benötigt. Somit kommt man auf vier Verbindungskabel. Zusätzlich müssen noch beide Massekontakte verbunden werden. Außerdem müssen zum Nachbau des Interface noch je ein Game-Port-Stecker und ein Userport-Stecker vorhanden sein. Der Zusammenbau sollte selbst für Nicht-Lötkolben-Schwinger einfach zu realisieren sein.
Wenn Sie das Interface an die Rechner anschließen, sollten Sie darauf achten, daß die Stekker richtig herum eingesteckt werden. Außerdem sollten beide Rechner ausgeschaltet sein!
Dem Schaltplan (Bild 1) können Sie entnehmen, welche Pins verbunden werden:
Verbindung 1 ist die Strobe-Ausgangsleitung des Apple zum Commodore. Beim Apple wird Annunciator 2 verwendet. Verbindung 2 ist der Daten-Ausgang des Apple II zum C-64. Genutzt wird Annunciator 1.
Der Datenbit-Eingang beim Apple ist der Pushbutton 1-Pin, der durch Leitung 3 mit dem C-64 verbunden ist.
Entsprechend ist Pushbutton 2 (Leitung 4) der Strobe-Eingang des Apple II.
Verbindung 5 verbindet Apple-Masse mit C-64-Masse.
Beim Apple II können Sie neben dem Assemblerlisting auch dem Reference-Manual die Speicherstellen entnehmen, über den Sie die serielle Schnittstelle steuern können (z. B. S. 131).
Beim Commodore werden die Standard-Ein/Ausgabeleitungen des Userports verwendet.
Mit dem Bau des Interface ist der erste Schritt zur Übertragung gemacht, der zweite besteht im Schreiben der geeigneten Softwaretreiber, die die Schnittstelle unterstützen.
Listing 1 ist ein Assemblerlisting der Treiber für den Apple, die C-64-Treibersoftware zeigt Listing 2.
Abgedruckt sind jeweils eine Routine zur Übertragung eines ASCII-Zeichens und zum Empfang eines solchen. Assemblerprogrammierern sollte es keine Schwierigkeiten bereiten, die Sourcecode-Listings der Routinen in eigene Assemblerprogramme einzubauen.
Aus den Assemblerlistings ist auch das Übertragungsprotokoll ersichtlich:
Ein Datenbyte besteht aus acht Bit, die nacheinander übertragen werden müssen. Da verschiedene Rechner nicht ohne weiteres im gleichen Takt senden und empfangen können, muß der Empfang eines Datenbits dem Sender bestätigt werden.
Zunächst muß der übertragende Computer dem Datenbit entsprechend die Datenleitung auf 0 oder 1 schalten. Anschließend wird das Bit der Strobeleitung (Signalleitung) dieses Rechners gesetzt, um dem Empfänger zu signalisieren, daß er die Datenleitung nun abfragen kann. Der Empfänger wartet also, bis das Strobebit des Senders auf eins gesetzt wurde. Dann liest er seine Dateneingangsleitung und speichert das empfangene Bit.
Inzwischen ist die Senderoutine in eine Schleife gekommen, die solange das Strobe-Bit des Empfängers abfragt, bis dieses gesetzt wird. Der Empfänger bestätigt dadurch, daß das gesendete Bit verarbeitet wurde.
Nun löscht der Sender sein Strobe-Bit und bestätigt damit den Empfang der Nachricht des anderen.
Der Empfänger wartet auf dieses Löschen des Signalbits des Senders und löscht sein Strobebit dann ebenfalls. Nachdem der sendende Computer dieses Signal erhalten hat, beginnt die Sendeschleife erneut mit dem nächsten Datenbit.
Um Übertragungsfehler zu ver-

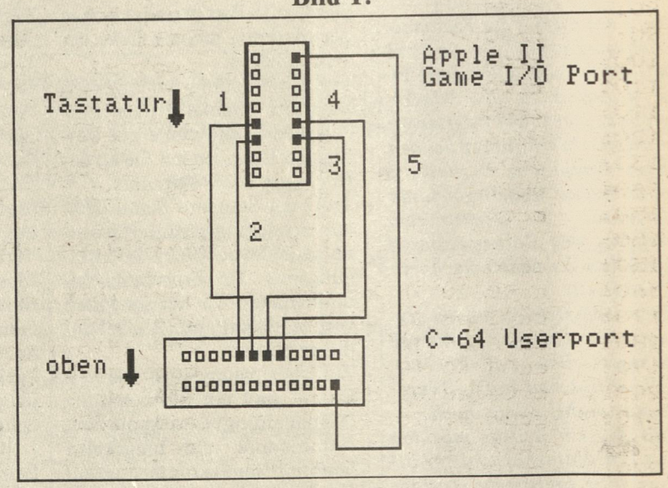
meiden, muß der sendende Computer vor dem Empfänger mit der Übertragung des ersten Bytes beginnen.
Die bisherigen Informationen zu Übertragungsprotokoll und Maschinenroutinen waren hauptsächlich für Assembler-Kenner bestimmt.
Doch auch an die Basic-Programmierer wendet sich dieser Artikel mit dem Interface-Bauvorschlag.
Zur Unterstützung der Basic-Programmierung des Interface wurde eine kleine, einfache Basic-Erweiterung für die beiden Computer entwickelt.

Listing 3 zeigt die Apple-Basic-Erweiterung in Maschinencode, die mit dem Apple II - Systemmonitor eingegeben wird. Listing 4 ist die entsprechende C-64-Routine.
Die Erweiterungen bringen dem Basic-Programmierer je zwei neue Befehle.
Sowohl beim Commodore 64 als auch beim Apple II lauten diese:
&INPUT : Empfangen eines Bytes. Der Wert wird in Adresse 255 gespeichert und kann mit PEEK(255) abgefragt werden.
&PRINT : Übertragen des Inhalts von Adresse 255 auf den anderen Rechner. Vor der Übertragung des Bytes muß der Wert mit POKE 255,WERT gespeichert werden.
Zur Demonstration der Anwendung dieser neuen Befehle sind mit Listing 5 und 6 zwei Basic-Programme für die beiden Computer abgedruckt, die Datenfiles (Text/Sequentielle Files) auf den anderen Rechner übertragen. Diese Programme verwenden also die Basic-Erweiterungen.
Hiermit ist es auch durchaus möglich, Basic-Programme zu übertragen. Diese müssen nur in ein solches Datenfile geschrieben und dann übertragen werden. Beim Apple muß dann mit Hilfe des EXEC - Befehls das File mit den Programmzeilen geladen werden.
Anschließend muß nur noch die teilweise unterschiedliche Basic-Syntax angepaßt werden. In diesem Basic-Programm haben wir die Vereinbarung getroffen, daß die Übertragung nach dem Senden des ASCII-Wertes 0 (CHR$(0)) beendet wird. Die Basic-Erweiterung ist davon jedoch nicht abhängig, deshalb kann natürlich auch jedes andere Zeichen verwendet werden.

Listing 1:

```
**********************************
*                                *
*   Apple-Unterroutinen zum      *
*   übertragen und Empfangen     *
*   eines Bytes auf bzw. vom     *
*        Commodore 64            *
*                                *
*       (C) 1984/85  by          *
*  Oliver Steinmeier (Apple)     *
*  Klaus Schneider   (C-64)      *
*                                *
**********************************

**********************************
*        E Q U A T E S           *
**********************************

DATAOUT1        EQU  $C05B
DATAOUT0        EQU  $C05A
STRBOUT1        EQU  $C05D
STRBOUT0        EQU  $C05C
DATAIN          EQU  $C062
STRBIN          EQU  $C063

**********************************
*     SENDBYTE-ROUTINE           *
*  überträgt den Accu-Inhalt     *
*       auf den C-64             *
**********************************

SENDBYTE        LDX  #8          ;Anzahl der Bits
                BIT  STRBOUT0
RDDOWN          BIT  STRBIN      ;Strobe lesen
                BMI  RDDOWN      ;Strobe gesetzt, dann warten
SENDLOOP        ROL
                BCC  NOSET
                BIT  DATAOUT1    ;Data-Bit setzen
                BCS  SETSTRBE
NOSET           BIT  DATAOUT0    ;Data-Bit löschen
SETSTRBE        BIT  STRBOUT1    ;Strobe setzen
RDSTRBE         BIT  STRBIN      ;Strobe lesen
                BPL  RDSTRBE     ;Strobe gesetzt ?
                BIT  STRBOUT0    ;Strobe löschen
RDSTRBE1        BIT  STRBIN      ;Strobe lesen
                BMI  RDSTRBE1    ;Strobe gelöscht ?
                DEX
                BNE  SENDLOOP
                RTS
```

Bild 1:




```
**********************************
*     GETBYTE-ROUTINE            *
*  holt ein Byte vom C-64        *
*     in den Accumulator         *
**********************************

GETBYTE   LDX  #8          ;Anzahl der Bits
          BIT  STRBOUT0
GETLOOP   BIT  STRBIN      ;Strobe lesen
          BPL  GETLOOP     ;Strobe gesetzt ?
          BIT  DATAIN      ;Data-Bit lesen
          BMI  DATASET     ;Data-Bit gesetzt
          CLC
          BCC  STRBON
DATASET   SEC
STRBON    ROL
          BIT  STRBOUT1    ;Strobe setzen
RDSTRB2   BIT  STRBIN      ;Strobe lesen
          BMI  RDSTRB2
          BIT  STRBOUT0    ;Strobe löschen
          DEX
          BNE  GETLOOP
          RTS
```

Listing 2: Treiberroutine für C-64

```
                                        profi-ass 64 v2.0

80:     cc00                    .pag 55
90:     cc00                    .tit "(c) 1984/1985  O. Steinmeier & K. Schneider"
100:    cc00                    .opt p1
110:    cc00                    *=   $cc00
110:    cc00
120:    dd01            dr      =    56576+1 ; datenregister
130:    dd03            ddr     =    56576+3 ; datenrichtungregister
140:    00ff            store   =    255     ; enthaelt byte
150:    cc00
150:                    ; ************ byte von apple holen ************
150:    cc00
160:    cc00 20 7b cc getbyte   jsr  init    ; register setzen
170:    cc03 ad 01 dd getb1     lda  dr      ; datenreg. lesen
180:    cc06 29 10              and  #16     ; test auf apple strobe
190:    cc08 f0 f9              beq  getb1   ; null, dann warten
200:    cc0a ad 01 dd           lda  dr      ; datenreg. lesen
210:    cc0d 29 20              and  #32     ; datenbit isolieren
220:    cc0f 0a                 asl
220:    cc10 0a                 asl
220:    cc11 0a                 asl          ; und in carry bringen
230:    cc12 26 ff              rol  store   ; carry an byte haengen
240:    cc14 a9 04              lda  #4
250:    cc16 0d 01 dd           ora  dr
260:    cc19 8d 01 dd           sta  dr      ; c64 strobe setzen
270:    cc1c ad 01 dd getb2     lda  dr      ; datenreg. lesen
280:    cc1f 29 10              and  #16     ; applestrobe isolieren
290:    cc21 d0 f9              bne  getb2   ; noch 1, dann warten
300:    cc23 a9 fb              lda  #251
310:    cc25 2d 01 dd           and  dr
320:    cc28 8d 01 dd           sta  dr      ; c64 strobe loeschen
330:    cc2b ca                 dex          ; bitzaehler - 1
340:    cc2c d0 d5              bne  getb1   ; groesser null, dann naechstes bit
350:    cc2e f0 39              beq  return  ; null, dann ende
360:    cc30
360:                    ; ************ byte zum apple senden **********
360:    cc30
370:    cc30 20 7b cc sendbyte  jsr  init    ; register setzen
380:    cc33 a9 f3              lda  #$f3    ; bitmaske c64 strobe & data = 0
390:    cc35 2d 01 dd           and  dr      ; mit datenreg. verknuepfen
400:    cc38 8d 01 dd           sta  dr      ; und setzen
410:    cc3b 06 ff    sendb1    asl  store   ; ein bit ins carry
```

```
420:   cc3d ad 01 dd sendb2   lda  dr      ; datenreg lesen
430:   cc40 29 10             and  #16     ; applestrobe isolieren
440:   cc42 d0 f9             bne  sendb2  ; noch gesetzt, dann warten
450:   cc44 2a                rol
450:   cc45 2a                rol
450:   cc46 2a                rol
450:   cc47 2a                rol          ; carry in register schieben
460:   cc48 09 04             ora  #4      ; c64 strobe setzen
470:   cc4a 0d 01 dd          ora  dr
480:   cc4d 8d 01 dd          sta  dr      ; und senden
490:   cc50 ad 01 dd sendb3   lda  dr      ; datenreg. lesen
500:   cc53 29 10             and  #16     ; applestrobe isolieren
510:   cc55 f0 f9             beq  sendb3  ; noch nicht gesetzt
520:   cc57 a9 f3             lda  #$f3    ; maske fuer c64 strobe & data = 0
530:   cc59 2d 01 dd          and  dr      ; mit datenreg. verknuepfen
540:   cc5c 8d 01 dd          sta  dr      ; und speichern
550:   cc5f ad 01 dd sendb4   lda  dr      ; datenreg. lesen
560:   cc62 29 10             and  #16     ; applestrobe isolieren
570:   cc64 d0 f9             bne  sendb4  ; noch 1, dann warten
580:   cc66 ca                dex          ; bitzaehler - 1
590:   cc67 d0 d2             bne  sendb1  ; ungleich 0, dann naechstes bit
600:   cc69
600:                ; ********** register zuruecksetzen **********
600:   cc69
610:   cc69 a9 01    return    lda  #1      ; bit fuer irq
620:   cc6b 0d 0e dc          ora  56334   ; setzen
630:   cc6e 8d 0e dc          sta  56334   ; und speichern
640:   cc71 a2 00             ldx  #0
650:   cc73 8e 01 dd          stx  dr      ; datenreg. loeschen
660:   cc76 ca                dex          ; x = 255
670:   cc77 8e 03 dd          stx  ddr     ; alle bits auf ausgang
680:   cc7a 60                rts
690:   cc7b
690:                ; ************** register setzen ***************
690:   cc7b
700:   cc7b a9 fe    init      lda  #$fe
710:   cc7d 2d 0e dc          and  56334
720:   cc80 8d 0e dc          sta  56334   ; irq abschalten
730:   cc83 a9 00             lda  #0
740:   cc85 8d 01 dd          sta  dr      ; eventuelle alte daten loeschen
750:   cc88 a2 cf             ldx  #$cf    ; bit4 & bit5 eingabe, rest ausgabe
760:   cc8a 8d 03 dd          sta  ddr     ; ins datenrichtungregister
770:   cc8d a2 08             ldx  #8      ; zaehler fuer 8 bit
780:   cc8f 60                rts
]cc00-cc90
```

FILE 'DATENUEBERTRAGUNG' IST DAS BASIC- PROGRAMM ZUR TEXTFILE-UEBERTRAGUNG
FILE 'DATENUEBERTRAGUNG.BIN' IST DAS ZU-GEHOERIGE MASCHINENPROGRAMM

Listing 3: Basic-Erweiterung für den Apple II in Assembler

```
0300- A9 10 A2 03 A0 4C 8C F5
0308- 03 8D F6 03 8E F7 03 60
0310- C9 BA F0 0F C9 84 F0 03
0318- 4C C9 DE 20 54 03 85 FF
0320- 4C B1 00 A5 FF 20 2B 03
0328- 4C B1 00 A2 08 2C 5C C0
0330- 2C 63 C0 30 FB 2A 90 05
0338- 2C 5B C0 B0 03 2C 5A C0
0340- 2C 5D C0 2C 63 C0 10 FB
0348- 2C 5C C0 2C 63 C0 30 FB
0350- CA D0 E2 60 A2 08 2C 5C
0358- C0 2C 63 C0 10 FB 2C 62
0360- C0 30 03 18 90 01 38 2A
0368- 2C 5D C0 2C 63 C0 30 FB
0370- 2C 5C C0 CA D0 E3 60
```


Listing 4: BASIC-Erweiterung für C-64

```
100 REM ****************************************
110 REM *                                      *
120 REM *    DATENUEBERTRAGUNG C64/APPLE ][    *
130 REM *                                      *
140 REM *        COPYRIGHT 1984/1985 BY        *
150 REM *                                      *
160 REM *            OLIVER STEINMEIER         *
170 REM *                                      *
180 REM *         UND KLAUS SCHNEIDER          *
190 REM *                                      *
200 REM *      BASICERWEITERUNG FUER C64       *
210 REM *                                      *
220 REM ****************************************
230 :
240 FOR I = 52224 TO 52423
250   READ A
260   POKE I,A
270   CO = CO + A
280 NEXT I
290 IF CO < > 23254 THEN PRINT "ERROR IN DATAS!" : END
300 PRINT "ALLES OK!"
310 SYS 52224
320 END
330 :
340 DATA 169,116,141,008,003,169,204,141,009,003
350 DATA 096,162,008,173,001,221,041,016,240,249
360 DATA 173,001,221,041,032,010,010,010,038,255
370 DATA 169,004,013,001,221,141,001,221,173,001
380 DATA 221,041,016,208,249,169,251,045,001,221
390 DATA 141,001,221,202,208,213,240,117,162,008
400 DATA 006,255,169,243,045,001,221,141,001,221
410 DATA 173,001,221,041,016,208,249,042,042,042
420 DATA 042,009,004,013,001,221,141,001,221,173
430 DATA 001,221,041,016,240,249,169,243,045,001
440 DATA 221,141,001,221,173,001,221,041,016,208
450 DATA 249,202,208,202,240,059,032,115,000,201
460 DATA 038,240,006,032,121,000,076,231,167,032
470 DATA 115,000,201,133,240,007,201,153,240,003
480 DATA 076,008,175,133,002,169,254,045,014,220
490 DATA 141,014,220,169,000,141,001,221,169,207
500 DATA 141,003,221,165,002,201,133,240,003,076
510 DATA 058,204,076,011,204,169,001,013,014,220
520 DATA 141,014,220,162,000,142,001,221,202,142
530 DATA 003,221,032,115,000,076,174,167,234,234
```

Listing 5: Apple-Programm zur Textfile-Übertragung

```
10  REM ************************************
20  REM *                                  *
30  REM *    Datenübertragung zwischen     *
40  REM *                                  *
50  REM *   Apple II und Commodore 64      *
60  REM *                                  *
70  REM *                                  *
80  REM *  O. Steinmeier & K. Schneider    *
90  REM ************************************
```

```
95  PRINT : PRINT  CHR$ (4);"BLOAD DATENUEBERTRAGUNG.BIN"
96  CALL 768
100  REM   ****   Menue   ****

110  HOME : INVERSE : PRINT "    D A T E N U E B E R T R A G U N G
     ": NORMAL
120  VTAB 5: PRINT "Menue : "
130  VTAB 8: PRINT "<1>  Textfile auf C-64 übertragen"
140  VTAB 10: PRINT "<2>  Textfile von C-64 empfangen"
150  VTAB 12: PRINT "<3>  Ende"
160  VTAB 16: PRINT "Bitte wählen : ";
170  GET IP$: IF IP$ > "3" OR IP$ < "1" THEN 170
180  ON  VAL (IP$) GOSUB 1000,2000,3000
190  GOTO 100
500  REM  Fileparameter eingeben
510  VTAB 5: INPUT "Filename des Textfiles : ";N$
520  VTAB 7: INPUT "Slot : ";S$
530  VTAB 9: INPUT "Drive : ";D$
540 F$ = N$ + ",S" + S$ + ",D" + D$
560  VTAB 14: PRINT  TAB( 12);"Bitte warten ..."
570  RETURN
1000  REM   **** Textfile auf C-64 übertragen  ****
1010  HOME : INVERSE : PRINT "      TEXTFILE AUF C-64 ÜBERTRAGEN
     ": NORMAL
1020  GOSUB 500: REM  Eingabe der Fileparameter
1025 S = 0
1030  ONERR  GOTO 1200
1040  PRINT : PRINT  CHR$ (4);"OPEN";F$
1050  PRINT  CHR$ (4);"READ";N$
1055  VTAB 18: PRINT "Bitte auf dem C 64 <RETURN> drücken !"
1060 S = S + 1: GET A$: POKE 255, ASC (A$): &  PRINT
1070  GOTO 1060
1200  POKE 216,0: REM     ON ERR abschalten
1205  PRINT : PRINT  CHR$ (4);"CLOSE"
1210  POKE 255,0: &  PRINT : REM   END OF TEXT ERREICHT
1220  IF  PEEK (222) <  > 5 THEN  VTAB 20: PRINT  CHR$ (7);"Fehler "
    ; PEEK (222);" ist aufgetreten !";: GOTO 1260
1230  VTAB 20: PRINT "Kein Fehler bei der übertragung !"
1240  PRINT : PRINT "Es wurden ";S;" Byte übertragen !"
1250  PRINT : PRINT  TAB( 10);"Taste drücken : ";
1260  GET IP$: RUN 100
2000  REM  **** Textfile von C-64 empfangen ****
2010  HOME : INVERSE : PRINT "    TEXTFILE VON C-64 EMPFANGEN
     ": NORMAL
2020  GOSUB 500: REM  Eingabe der Fileparameter
2023  VTAB 16: INPUT "Bitte <RETURN> drücken, sobald der C 64 dazu a
    uffordert ! ";IP$
2025 S = 0
2030  PRINT : PRINT  CHR$ (4);"OPEN";F$
2040  PRINT  CHR$ (4);"DELETE";N$
2050  PRINT  CHR$ (4);"OPEN";N$
```



```
2060  PRINT  CHR$ (4);"WRITE";N$
2070  &  INPUT : IF  PEEK (255) = 0 THEN 2090
2080  PRINT  CHR$ ( PEEK (255));:S = S + 1: GOTO 2070
2090  PRINT  CHR$ (4);"CLOSE"
2100  PRINT : PRINT "Es wurden ";S;" Byte übertragen"
2110  PRINT : PRINT  TAB( 10);"Taste drücken : ";: GET IP$
2120  RETURN
3000  REM  **** Ende ****
3010  HOME : VTAB 12: PRINT  TAB( 18);"Ende ? ";: GET IP$
3020  IF IP$ = "J" OR IP$ = "j" THEN  HOME : PRINT "Tschüß ": VTAB 2
    0: END
3030  RETURN
```

Listing 6: C-64-Programm zur Übertragung eines Textfiles

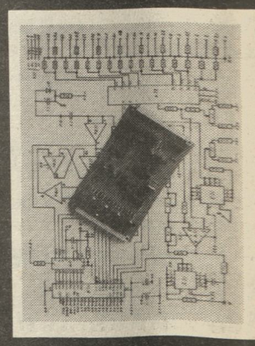
```
100 rem ****************************************
110 rem *                                        *
120 rem *      datenuebertragung zwischen        *
130 rem *                                        *
140 rem *      apple ][ und commodore 64         *
150 rem *                                        *
160 rem *                                        *
170 rem *   k. schneider und o. steinmeier       *
180 rem *                                        *
190 rem ****************************************
200 :
210 rem *************** menue **************
220 :
230 print chr$(14)chr$(8)"■■■   D a t e n u e b e r t r a g u n g    "
240 print "■■■■"
250 print " <1>  Textfile auf Apple II uebertragen"
260 print "■■"
270 print " <2>  Textfile von Apple II empfangen"
280 print "■■"
290 print " <3>  Ende"
300 get ip$ : if ip$ < "1" or ip$ > "3" then 300
310 on val(ip$) gosub 560,910,1120
320 goto 230
330 :
340 rem ******* fileparameter eingeben ******
350 :
360 input "■■ Filename ";n$
370 print
380 input " Geraetenummer ";dv%
390 print
400 input " Sekundaeradresse ";sn%
410 print
420 print " Soll eine C64/ASCII Code Umwandlung "
430 print
440 print " erfolgen (J/N) ? ";
450 get c$ : if c$ < > "j" and c$ < > "n" then 450
460 print "JA■■■";
470 if c$ = "n" then print "NEIN";
```

```
480 print
490 print
500 print " Bitte Uebertragung vorbereiten und "
510 print
520 print " dann <RETURN> druecken!"
530 get ip$ : if ip$ < > chr$(13) then 530
540 return
550 :
560 rem *** textfile auf apple uebertragen ***
570 :
580 gosub 340 : rem eingabe der fileparameter
590 if dv% < 4 or dv% > 6 then n$ = n$ + ",s,r"
600 s = - 1
610 open 1,dv%,sn%,n$
620 get #1,a$
630 print
640 print " Bitte auf dem Apple <RETURN> druecken!"
650 goto 670
660 get #1,a$
670 if a$ = "■" or a$ = "" then 660
680 if c$ = "j" then gosub 820
690 if st < > 0 then a$ = chr$(0)
700 poke 255,asc(a$)
710 &print
720 s = s + 1
730 if st = 0 then 660
740 close 1
750 print
760 print " Es wurden";s;"Byte uebertragen!"
770 print
780 print " Taste druecken !"
790 get ip$ : if ip$ = "" then 790
800 run
810 :
820 rem ******* ascii/c 64 umwandlung ******
830 :
840 a1 = asc(a$)
850 a2 = a1 and 127
860 if a2 > 64 and a2 < 91 then a1 = a2 + 32:goto 880
870 if a2 > 96 and a2 < 123 then a1 = a1 - 32
880 a$ = chr$(a1)
890 return
900 :
910 rem *** textfile von apple empfangen ***
920 :
930 gosub 340 : rem eingabe der fileparameter
940 if dv% < 4 or dv% > 6 then n$ = n$ + ",s,w"
950 open 1,dv%,sn%,n$
960 s = 0
970 &input
980 s = s + 1
990 a$ = chr$(peek(255))
1000 if c$ = "j" then gosub 820
```

- VORSCHAU -

In der nächsten Tips u. Tricks von CPU"!

- Start eines mehrteiligen Spectrum-Kurses: Professionelles Programmieren in Maschinensprache. Neben ausführlichen Erläuterungen wird ein Topprogramm für den Spectrum-Computer entwickelt.
- Aufgrund zahlreicher Leserzuschriften beginnen wir mit einer neuen 3-teiligen Serie: - VC-20-/C-64-Betriebssystem -
- Fortsetzung der Hardware-Serie: „Der Commodore als Steuerungscomputer" (siehe Foto!)
- Wieder zahlreiche Programme und Beiträge auch für Ihr Computersystem u. a.
  - C-64 als Speicheroszilloskop
  - Interrupt-Uhr für den Schneider 464
  - u.v.m.
- Aktueller Softwaretest: TURBO-PASCAL jetzt auch für den Schneider CPC 464
- Fortsetzung unseres Autorenwettbewerbs: „Wir" suchen wieder die besten Beiträge für: Tips u. Tricks; User und Utility



Schaltplan und Platine zur Hardware-Serie


```
1010 print#1,a$;
1020 if peek(255) < > 0 then 970
1030 print#1
1040 close 1
1050 print
1060 print " Es wurden";s;"Byte uebertragen!"
1070 print
1080 print " Taste druecken !"
1090 get ip$ : if ip$ = "" then 1090
1100 run
1110 :
1120 rem ***************** ende ****************
1130 :
1140 print "■ Ende (J/N) ?"
1150 get ip$ : if ip$ < > "n" and ip$ < > "j" then 1150
1160 if ip$ = "j" then print "■Tschuess!!" : end
1170 return
```

ERROR-Helper für den ATARI 800 XL (Teil 2)

Funktion und Aufbau dieses Programms ist bereits in „Tips und Tricks von CPU" Nr. 9 ausführlich erläutert worden!

```
0 DIM NUMMER$(5),FILE$(13),LESE$(256):TRAP 30000:OPEN #2,4,0
,"K"
30000 IF TRAP=0 THEN 30020
30010 LET TRAP2=TRAP:LET TRAP=0:GOTO TRAP2
30020 GRAPHICS 0:? "■":POSITION 0,0:? "■ERROR-HELPER by GUI
do Michel (C)1985 ":DLHELP=PEEK(560)+PEEK(561)*256+4
30030 POKE DLHELP+2,112
30040 NUMMER$=STR$(PEEK(195)):ZEILE=PEEK(186)+PEEK(187)*256:
FILE$="D:FEHLER.":FILE$(LEN(FILE$)+1)=NUMMER$
30050 ? :? "FEHLER ";NUMMER$;" IN ZEILE ";ZEILE
30060 TRAP 30070:CLOSE #1:OPEN #1,4,0,FILE$:INPUT #1;LESE$:?
 "↓↓";LESE$:GET #2,TASTE:GOTO 30080
30070 POSITION 2,5:? "Diskette mit Fehler ";NUMMER$;" ein-":
? "legen & RETURN druecken":GET #2,TASTE:GOTO 30060
30080 TRAP 40000:NZEILE=0:ADRESSE=PEEK(136)+PEEK(137)*256
30090 BZEILE=PEEK(ADRESSE)+PEEK(ADRESSE+1)*256
30100 IF NZEILE=1 THEN NZEILE=BZEILE:GOTO 30140
30110 IF BZEILE=ZEILE THEN NZEILE=1
30120 ADRESSE=ADRESSE+PEEK(ADRESSE+2)
30130 GOTO 30090
30140 GRAPHICS 0:POKE DLHELP+2,112:POSITION 0,0:? "■ERROR-H
ELPER by Guido Michel (C)1985 "
30150 ? "↓Druecke Anfangsbuchstaben um..."
30160 ? "↓die Fehlerzeile zu ■isten.":? "↓das Programm zu St
oppen.":? "↓mit naechster Zeile ■ortzufahren."
30170 ? "↓um mit der Fehlerzeile ■eiterzumachen"
30180 GET #2,TASTE
30190 IF TASTE=76 THEN 30240
30200 IF TASTE=83 THEN 30250
30210 IF TASTE=70 THEN 30260
30220 IF TASTE=87 THEN 30270
30230 GOTO 30140
30240 GRAPHICS 0:? :LIST ZEILE:END
30250 GRAPHICS 0:? :STOP
30260 GRAPHICS 0:GOTO NZEILE
30270 GRAPHICS 0:GOTO ZEILE
```

LISTING 1
10 TRAP 100 : OPEN "1,4,0, "D:"
UMGEÄNDERTES LISTING
10 LET TRAP = 100 ; OPEN "1,4,0, "D:"

Tips und Tricks für Computer des MSX-Standards

Spriteposition in Bildschirmkoordinaten

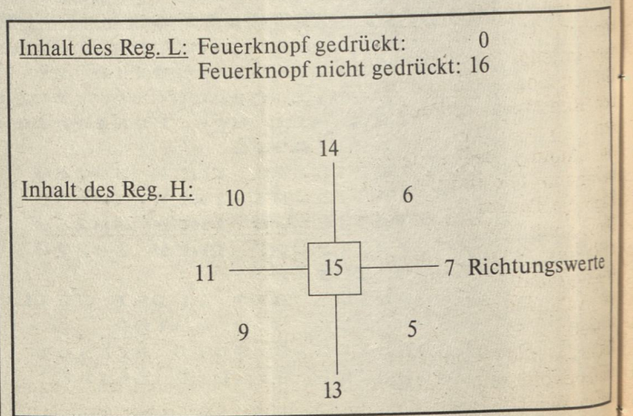
Beschreibung der Maschinenroutine „SPRPOS“

Die Funktion der Routine besteht darin, daß die momentane Position eines bestimmten Sprites in eine Bildschirmposition (Feld) für den Textmodos 2 (Screen 1) umgerechnet wird. Dadurch ist es z. B. möglich, die Kollision zwischen Sprite's und einem bestimmten Zeichen aus dem Bildschirm zu überprüfen. (Der VDP besitzt nämlich kein Register das dies anzeigt).
Die Routine ist in den Datazeilen 50 bis 110 enthalten und benötigt keine bestimmten Speicheradressen; d. h. sie kann in beliebige (vom Basic nicht belegte) Adressen geschrieben werden (am besten ist es, sich mit der CLEAR-Anweisung entsprechenden Platz zu schaffen). Wird die Routine mit DEF USR 0 definiert, so wird sie im Zusammenhang mit einer beliebigen Variable (z. B. A) auch mit A=USR0(0) aufgerufen. Die Zahl in der Klammer (keine Variable!) bestimmt das Sprite (0 - 31). Nach Aufruf der Routine wird der Variable (im Demo A) die Bildschirmposition des ausgewählten Sprite's übergeben, und so kann diese dann für weitere Zwecke verwendet werden. Die Zeilen 130 bis 240 im Basicprogramm dienen als Demonstration.
Für Interressierte: Der Basic-Interpreter der MSX-Rechner verwendet die Speicherstellen 63480 und 63481 (LSB, MSB) zum beiderseitigen Datentransfer zwischen Basic und Maschinensprachroutinen.

```
10 REM*** SPRITEPOSITIONSROUTINE ***
20 CLEAR100,60416!
30 SCREEN1,0
40 FORY=60416!TO60472!:READA:POKEY,A:NEXTY
50 DATA58,248,247,33,0,27,7,7,22,0
60 DATA95,25,205,74,0,198,3,95,35
70 DATA205,74,0,198,3,87,175,122
80 DATA31,31,31,230,31,111,175,123
90 DATA15,15,15,15,15,15,79,230,3
100 DATA246,24,103,121,230,224,181
110 DATA111,237,99,248,247,201
120 DEFUSR0=60416!
130 REM*** DEMO ***
140 FORY=0TO7:READA:A$=A$+CHR$(A):NEXTY
150 DATA0,0,24,60,60,24,0,0
160 SPRITE$(0)=A$:VPOKE6470,202:VPOKE6489,
    200:Y=80:X=56:X1=1:F=1
170 LOCATE5,5:PRINT"DEMONSTRATION"
180 LOCATE3,9:PRINT"6470":LOCATE22,9:PRINT"6489"
190 PUTSPRITE0,(X,Y),F,0
200 A=USR0(0):LOCATE6,12:PRINT"SPRITEPOS.:";A
210 IFVPEEK(A)=202THENX1=1:F=1
220 IFVPEEK(A)=200THENX1=-1:F=6
230 X=X+X1:GOTO190
240 END
```

Beschreibung der Maschinensprachroutine „JOY“

Diese Routine ist für User gedacht, die nur in Maschinensprache programmieren wollen und dient dazu die beiden Joystickports abzufragen. Die Kontrolle über die Port's obliegt dem Soundchip (AY-3-8912) mit seinen Registern 14 und 15 (die als Ein- und Ausgabeport's fungieren). Vor dem Einsprung in die Routine ist das Register H des Z80-Prozessors mit einem Wert zu laden, der bestimmt, welcher Joystick abgefragt wird. Dies ist im Basic-Listing in der Datazeile 100 ersichtlich, die nicht zur Routine gehört, sondern nur für die Demonstration notwendig ist. (Der dezimale Wert 129 ist für Joystick 1, der Wert 196 für Joystick 2).
Die eigentliche Routine steht in den Datazeilen 40 bis 60, der Rest des Programmes dient zur Demonstration. In der Routine wird dann der Soundchip mit den Z80-Befehlen OUT und IN angesprochen und liefert schließlich die entsprechenden Werte.
Mit den logischen Funktionen wird noch die Bewegungsrichtung und der Feuerknopf voneinander getrennt, um dem Register H den Richtungswert, und dem Register L den Zustand des Feuerknopfes zu übertragen. (Auch diese Routine ist Adressenkompatibel).




```
10 REM*** JOYSTICKROUTINE ***
20 CLEAR100,60416!
30 FORY=60416!TO60437!:READA:POKEY,A:NEXTY
40 DATA62,15,211,160,124,211,161,62
50 DATA14,211,160,219,162,103,230
60 DATA16,111,124,230,15,103,201
70 REM*** DEMO ***
80 CLS
90 FORY=60438!TO60449!:READA:POKEY,A:NEXTY
100 DATA38,129,205,0,236,237,99,32,236,201,0,0:REM
                                (129=JOY1;196=JOY2)
110 DEFUSR0=60438!
120 X=USR(0)
130 F=PEEK(60448!):R=PEEK(60449!)
140 LOCATE5,5:PRINT"RICHTUNG:";R
150 LOCATE5,9:PRINT"FEUERKNP:";F
160 GOTO120
170 END
```

Erweiterung des MSX-Befehlssatzes um zwei Grafikbefehle

Die zwei vorliegenden Programme erweitern den MSX-Befehlssatz um zwei Grafikbefehle. Beide Befehle wurden den analogen Ausdrücken des Simons Basic (lauffähig auf dem C 64) nachempfunden und stellen dieselbe Anzahl von Parametern, die frei verändert werden können, zur Verfügung.

I. Der ARC-Befehl

Der Arc-Befehl (vergl. arc de triumph) erlaubt es, wie der Name schon sagt, alle möglichen Arten von Bögen und Vielecken im High-Resolution Modus darzustellen. Im Klartext: Es können Dreiecke, Vierecke usw. in jeder Position, in allen Größen und Ausschnitten auf den Bildschirm gebracht werden.
Am Anfang des Programms werden mittels einer Input-Abfrage die Parameter festgelegt. Diese sind:
r (Radius), x (x-Koordinate), y (y-Koordinate), e (Eckenzahl), aw (Anfangswinkel), ew (Endwinkel).
Radius, x- und y-Koordinate bedürfen wohl keiner Erklärung. Anfangswinkel steht für die Gradzahl (0-360), bei der die Figur beginnen soll. Endwinkel steht für die Gradzahl bei der die Figur endet. Eine vollständige Figut erhält man mit den Werten aw=0 und ew=360. Diese beiden Variablen begrenzen also den Ausschnitt der darzustellenden Figur. Mit Eckenzahl ist die Anzahl der Ecken gemeint, die sich ergibt, wenn man eine vollständige Figur darstellt.

II. Der ANGLE-Befehl

Der Angle-Befehl beruht größtenteils auf dem Arc-Befehl. Er dient dazu einen, oder mehrere Radien in einen Kreis einzuzeichnen. Der Kreis muß dazu nicht dargestellt werden. Bildlich gesehen wird also eine Linie, oder mehrere Linien vom Mittelpunkt des Kreises zum Kreisbogen gezeichnet. Die Parameter r,x,y,aw und ew werden genau wie bei I. gehandhabt. Anstatt Eckenzahl wird aber nun Radienzahl benutzt.

Beispiel: Bei einer Radienzahl von 4 werden, vorausgesetzt aw=0 und ew=360, im Winkel von 90 Grad vier Radien gezeichnet (360/4=90). Die abgedruckten Programme zeichnen, bei MSX-Computern, Figuren, mit einer übermäßigen x-Ausdehnung von ungefähr ein Drittel. D.h. es werden Ellypsen dargestellt, deren x-Radius etwa um ein Drittel größer ist als der y-Radius. Das kann man ändern, indem man die Definitionszeilen von y1, und beim Arc-Befehl noch y2, wie folgt ändert:
100 y1=r·1.4 sin(b/a)+y
und beim Arc-Befehl noch die Zeile
120 y2=r·1.4 sin((p+b)/a)+y
Anbei noch eine Demonstration für den Arc-Befehl. Die folgenden Zeilennummern müssen beibehalten werden:
75 for p=10 to 360 step 5
145 next p
Die Eingabe von e kann bei diesem Beispiel entfallen, da es vom Programm vorgegeben wird.

Hinweis:
Dieses Programm ist auf die meisten Computer mit Grafikunterstützung leicht umzuschreiben. Vorausgesetzt wird lediglich der Line-Befehl.

```
1  '**********************************
2  '**********************************
3  '**                              **
4  '**         Printed by           **
5  '**                              **
6  '** Peter Brill & Stephan Paul   **
7  '**                              **
8  '**********************************
9  '**********************************
10 CLS
20 INPUT"RADIUS,X,Y,ECKZAHL,AW,EW";R,X,Y,E,AW,EW
30 P=360/E
40 SCREEN2
50 A=57.2958
60 V=AW-90
70 W=EW-90
80 FOR B=V TO W STEP P
90 X1=R*COS(B/A)+X
100 Y1=R*1.4*SIN(B/A)+Y
110 X2=R*COS((P+B)/A)+X
120 Y2=R*1.4*SIN((P+B)/A)+Y
130 LINE(X1,Y1)-(X2,Y2),15
140 NEXTB
150 GOTO150
```

```
10 CLS
20 INPUT"RADIUS,X,Y,RADIENZAHL,AW,EW";R,X,Y,RZ,AW,EW
30 P=360/RZ
40 SCREEN2
50 A=57.2958
60 V=AW-90
70 W=EW-90
80 FOR B=V TO W STEP P
90 X1=R*COS(B/A)+X
100 Y1=R*1.4*SIN(B/A)+Y
110 LINE(X1,Y1)-(X,Y),15
120 NEXTB
130 GOTO130
```

ACHTUNG!

Korrektur zum Spectrum-Programm „Variable List“ aus der CPU Nr. 6

Wir möchten uns an dieser Stelle bei unserem aufmerksamen Leser Stefan Rösch aus Singen bedanken, der uns auf den Fehler in Zeile 410 hinwies.

410>DATA ”23E53E0DD72115FFCD1CF
EE1CDB433E5CDE32D3E“

Tips und Tricks für den SVI-328

1. Zuerst einige Nachträge zum 1. Teil (CPU 9/85):

A) Tastaturabfrage
Von je einer Speicherzelle des RAM-Bereichs &HFD75 ... &HFD8A können 8 Tasten verwaltet werden, da jedes Bit eine Taste überprüft.
Speziell für die Adresse &HFD7B sieht das so aus:

Bit	7	6	5	4	3	2	1	0
Taste	←	ENTER	STOP	ESC	R. Grph	L. Grph	CTRL	SHIFT

Will man etwa wissen, ob die CTRL-Taste gedrückt ist, egal, ob auch noch andere Tasten gedrückt sind oder nicht, so erreicht man dies durch die Programmzeile

```
XXX IF PEEK(&HFD7-B)AND2 THEN XXX
```

In der Programmausführung wird erst dann fortgefahren, wenn die CTRL-Taste gedrückt ist.
Ein entsprechender Test für die SELECT-Taste lautet:

```
XXX IF PEEK(&HFD7-D)AND16 THEN XXX
```

Von der Adresse &HFD7D wird neben den genannten 7 Tasten auch die Taste ↓ (Cursor down) überprüft und von der Adresse &HFD7D zusätzlich die CAPS-LOCK-Taste.
Insgesamt werden also 87 Tasten abgefragt, also zwei weniger als der SVI-328 zur Verfügung hat. Dennoch werden natürlich alle Tasten überprüft, da ja ENTER und SHIFT doppelt vorkommen und in der Abfrage kein Unterschied gemacht wird, ob die rechte oder die linke SHIFT-Taste gedrückt wird, die rechte oder die linke ENTER-Taste.
B) Im Beispiel unter Punkt 4. wurde versehentlich die folgende Zeile vergessen:

```
:
100 PLAY A$,B$,C$
110 IF PLAY(0) THEN 110
:
```

2. Abfrage des programmierbaren Tongenerators (Programmable Sound Generator, PSG):

Wie im ersten Teil von „Tips und Tricks für den SVI-328“ bereits erwähnt, läßt sich mit IF PLAY(X), X=0,1,2 oder 3, abfragen, ob eine PLAY-Anweisung schon abgearbeitet wurde.
Man erhält jedoch keine Auskunft darüber, welcher Ton gerade gespielt wird. Dies ist jedoch durchaus möglich.
Zunächst aber eine Vorbemerkung zum besseren Verständnis:
Eine PLAY-Anweisung ist nichts anderes als eine interne Aneinanderreihung von bestimmten SOUND-Befehlen. Es werden, je nach zu spielender Note, 2 Bytes an den PSG übergeben, der daraus dann einen Ton einer bestimmten Frequenz erzeugt.
Die Register des Tongenerators lassen sich nicht nur von außen beschreiben, sondern auch lesen. Dafür ist jedoch kein spezieller BASIC-Befehl vorgesehen. Es ist aber trotzdem möglich, wenn auch über Umwege. Der Schlüssel hierzu sind die viel zu wenig beachteten OUT- und INP- Befehle, die an späterer Stelle nochmals ihre Leistungsfähigkeit demonstrieren. Hat man also einen Tonkanal (T) mit einem Datenbyte (X) beschrieben, zum Beispiel durch SOUND T,X , so läßt sich der Wert X später wieder abfragen durch die Befehlsfolge

```
OUT &H88,T : X=INP(&H90)
```

Beispiele:

```
SOUND 1,5 : ..... : OUT &H88,1: A=INP(&H90)
```

liefert für A den Wert 5 .

```
SOUND 8,12: ..... : OUT &H88,8: A=INP(&H90)
```

liefert für A die Lautstärke des Kanals A !
Diese Abfrage hat natürlich nur Zahlenwerte zum Ergebnis, keine Notennamen. Da jeder Note genau eine Zahlenkombination entspricht, kann man aus dieser auf die Note schließen.
Es folgt nun eine Tabelle, die jeder Note die beiden Datenbytes aus den entsprechenden Registern zuordnet und umgekehrt.
(Register 0/1 : Kanal A
Register 2/3 : Kanal B
Register 4/5 : Kanal C)

Die Abfrage

```
OUT &H88,0 : XO = INP(&H90)
OUT &H88,1 : X1 = INP(&H90)
```

ergibt stets die Werte der zuletzt gespielten Note, auch dann, wenn diese längst nicht mehr gespielt wird.

Doch Vorsicht:
Da der BEEP-Ton / CHR$(7) über Kanal A erzeugt wird, ist streng darauf zu achten, daß bei einer Abfrage des Tones von Kanal A/Register 0/1 es nicht irrtümlich zu einer Fehlinterpretation des Ergebnisses kommen kann. Die beiden anderen Tonkanäle bleiben unverändert.
Das Beschreiben der PSG-Register mittels OUT-Befehlen ist ebenso möglich.
SOUND 1,5 wird erreicht durch OUT&H88,1 : OUT &H8C,5 .
Dies ist vor allem für Maschinenprogramme wichtig, da der Z80-Befehl OUT dem BASIC-Befehl völlig gleich ist!!

3. Ein Tip für alle Disketten-Laufwerk-Besitzer

Laut Angaben des Herstellers darf eine Diskette erst dann aus dem Diskettenschacht genommen werden, wenn die LED-Anzeige des betreffenden Laufwerks erloschen ist, der Diskettenmotor also nicht mehr läuft. Der Motor aber ist nach jeder Diskettenoperation noch circa eine halbe Minute in Betrieb.
Muß der Besitzer eines Laufwerkes SVI-605 (nur ein Laufwerk) etwa beim Kopieren bestimmter Programme oder Dateien von einer Diskete auf eine andere öfters diese Disketten wechseln, so muß er „ewig“ warten, bis er seinen Kopiervorgang fortsetzen kann, will er nicht das Risiko eingehen, die Diskette zu zerstören, wenn er sie bei laufendem Motor aus dem Schacht nimmt.
Leider gibt es trotz des umfangreichen BASIC-Befehl-Satzes des SVI-328 keinen speziellen Befehl zum Ausschalten des Disk-Motors. Es ist trotzdem möglich, und zwar wieder mit einem OUT-Befehl:

```
OUT 52 , 0
```

Hiermit werden die Motoren beider Laufwerke abgeschaltet. Es lassen sich auch einige Spielereien damit betreiben, zum Beispiel:
Motor ausschalten, aber LED brennt weiter mittels OUT 52,1 (betrifft Laufwerk 1) bzw. OUT 52,2 : Motor 1 und 2 werden ausgeschaltet, ebenso LED 1, LED 2 aber brennt.
Motor einschalten (von Laufwerk 1) aber LED dabei nicht leuchten lassen, wird erreicht durch OUT 52,4 .
Eine weitere nützliche Anwendung ist die folgende:

Bei Programmen, die längere Zeit benötigen, und bei denen nur das Ergebnis wichtig ist, also nicht das, was zwischendurch passiert, kann man natürlich nicht immer den Bildschirm beobachten, ob das Programm nun endlich zu Ende ist. Ist der Computer zudem an einem Fernsehgerät angeschlossen, so schaltet man eben kurzerhand auf Fernsehempfang

um. Doch dann weiß man nicht, wann das Programm zu Ende ist. Man hat die Möglichkeit an das Programmende folgende Zeilen anzuhängen:

```
60000 INTERVAL ON
60010 ON INTERVAL
      = 30 GOSUB 60030
60020 GOTO 60020
60030 ' LED's beider
      Laufwerke ein-
      schalten
60040 OUT 52,3
60050 ON INTERVAL =
      30 GOSUB 60070
60060 RETURN
60070 'Beide LED's
      ausschalten
60080 OUT 52,0
60090 RETURN 60010
```

Dieses kleine Programm bewirkt, daß die beiden LED's der beiden Laufwerke blinken, hier, um den Benutzer zu signalisieren, daß das Programm abgearbeitet ist.

Note	Low-Byte (Reg. 0/2/4)								High-Byte (Reg. 1/3/5)							
c	232	116	186	221	239	119	60	30	14	7	3	1	0	0	0	0
c+	18	9	133	194	225	113	56	28	14	7	3	1	0	0	0	0
d	72	164	82	169	213	106	53	27	13	6	3	1	0	0	0	0
d+	137	69	34	145	201	100	50	25	12	6	3	1	0	0	0	0
e	213	235	245	123	189	95	47	24	11	5	2	1	0	0	0	0
f	43	150	203	101	179	89	45	22	11	5	2	1	0	0	0	0
f+	138	69	163	81	169	84	42	21	10	5	2	1	0	0	0	0
g	243	250	125	62	159	80	40	20	9	4	2	1	0	0	0	0
g+	100	178	89	45	150	75	38	19	9	4	2	1	0	0	0	0
a	221	111	55	28	142	71	35	18	8	4	2	1	0	0	0	0
a+	94	47	24	12	134	67	33	17	8	4	2	1	0	0	0	0
b	230	243	250	253	126	63	32	16	7	3	1	0	0	0	0	0
	Oktave 1	Oktave 2	Oktave 3	Oktave 4	Oktave 5	Oktave 6	Oktave 7	Oktave 8	Oktave 1	Oktave 2	Oktave 3	Oktave 4	Oktave 5	Oktave 6	Oktave 7	Oktave 8

Tips u. Tricks für den CPC 464

CP/M angepaßt

Wer das Diskettenlaufwerk DDI-1 für den Schneider CPC-464 kauft, bekommt das Betriebssystem CP/M 2.2 auf Diskette gratis mitgeliefert. Das Handbuch dazu ist schon äußerst mager ausgefallen; besonders schlimm schaut es aber bei SETUP.COM aus, das nur kurz angesprochen wird, obwohl sich damit CP/M an die eigenen Bedürfnisse anpassen läßt. Im folgenden Artikel wird die Bedienung des Programms erklärt. Vielleicht wollten Sie im CP/M-Modus schon einmal die Funktionstasten belegen, die Bildschirmfarben für den Farbmonitor ändern oder auch die Diskettenansteuerung z. B. an 5.25-Zoll-Laufwerke anpassen – all dies geht mit SETUP.COM. Wollen Sie SETUP laden, müssen Sie zuerst mit dem RSX-Befehl !CPM das CP/M-Betriebssystem laden. Aber Achtung: Bei diesem Bootvorgang werden alle Basic-Programme und Daten im Speicher gelöscht! Sobald die Startmeldung und der Cursor erscheinen, geben Sie SETUP und Enter ein. Das Programm meldet sich mit 'SETUP V2.0'. Der Reihe nach werden die gewünschten Systemeigenschaften abgefragt:

INITIAL COMMAND BUFFER:

Is this correct (Y/N):
Geben Sie 'N' ein und es erscheint 'Enter new initial command buffer:'. Hier können Sie einen Text eingeben, der beim Kaltstart vom CP/M als Befehl interpretiert und sofort ausgeführt wird. Eine Möglichkeit wäre die automatische Ausgabe des Disketten-Inhaltsverzeichnisses: DIR↑M
An diesem Beispiel sehen Sie auch, wie die ASCII-Steuerzeichen mit den Codes 0 bis 31 eingegeben werden: Mit vorangestelltem Potenzierungspfeil und dann der Taste ohne den Control-Wert. Beispiele hierfür sind:
↑M Return
↑J Line-Feed (ASCII-Code 10)
↑L Bildschirm löschen
(ASCII-Code 12)

SIGN-ON-STRING

Weiter geht es mit dem 'Sign-On-String': Normalerweise meldet sich CP/M mit der Copyright-Meldung von Amstrad Consumer Electronics. Falls Sie das stört, können Sie Ihrer Diskette eine 'persönliche Note' verpassen, z. B. 'Stets zu Diensten, Herr und Meister' oder ähnliches. Wichtiger hingegen sind die Steuerzeichen, die zusammen mit dem Text angegeben werden müssen. Sie ändern den Bildschirmmodus auf MODE 2 und setzen die Bildschirmfarben. Und damit wären wir bei einem wichtigen Punkt: Zumindest bei Einsatz des Farbmonitors belasten die von Amstrad gewählten Farben die Augen doch recht intensiv. Besser ist z. B. folgende Farbkombination:
↑i@JJ↑ia@@↑]JJCP/M 2.2 – Guten Tag↑J↑M

PRINTER POWER-UP STRING

Der nächste Programmpunkt ist der 'Printer power-up string'. Hiermit können beim Laden von CP/M automatisch Texte an den Drucker geschickt werden, z. B. zum Umschalten auf Schönschrift oder den deutschen Zeichensatz. Allerdings ist das etwas gefährlich, wenn der Drucker nicht eingeschaltet ist; denn der Computer wartet dann und man könnte glauben, er wäre ausgestiegen.

KEYBOARD TRANSLATION SET

Wichtiger ist hingegen das 'Keyboard translation set'. Hier können Sie – ähnlich dem KEY DEF-Befehl in Basic – die Tastatur den eigenen Schreibgewohnheiten anpassen. Das Programm zeigt ein kleines Menü:
Enter required command from:
A – Add key translation
D – Delete key translation
C – Clear all translations
F – Finish translations
Sie können z. B. entsprechend der deutschen Schreibmaschinenbelegung die Tasten 'Y' und 'Z' austauschen:
A 43,122,90,26
A 71,121,89,225
Wichtig wäre es auch, den Code der CLR-Taste zu verändern. Normalerweise hat sie die gleiche Funktion von Control-P. CP/M-Kenner wissen schon, was das heißt: Ein versehentliches Drücken der CLR-Taste, und alle Bildschirmausgaben laufen über den Drucker. Ist keiner angeschlossen, so steigt CP/M aus. Abhilfe schafft da:
A 16,8,8,12
Ab sofort fungiert CLR und Shift-CLR als Rückschritt-Taste und ist damit gleichwertig mit der danebenliegenden DEL-Taste. Beim Drücken von Control und CLR wird der Code von Control-L erzeugt, was den Bildschirm löscht.
Wollen Sie eine alte Tastendefinition wieder löschen, so geben Sie z. B. den Befehl 'D 1' ein.

Alle Neudefinitionen weden mit der C-Option wieder gelöscht; geben Sie also lieber gleich 'F' ('Finish' wie 'Beenden') ein. Es werden nochmals alle Definitionen aufgelistet.

KEYBOARD EXPANSION SET

Genauso interessant ist auch das Unterprogramm 'Keyboard expansion strings': Hier können Sie - wie mit KEY in Basic - die Funktionstasten mit Texten belegen. Eine nützliche Belegung ist z. B. folgende:

A 0 DIR↑M
A 1 ERA
A 2 REN
A 3 TYPE
A 4 USER
A 5 STAT
A 6 PIP
A 7 AMSDOS↑M

Damit belegen Sie den Zehnerblock mit den Hauptfunktionen von CP/M. Beim Drücken der Taste 7 im Zehnerblock springt der Computer ins Basic zurück – aber nur dann, wenn sich AMSDOS.COM auf der eingelegten Diskette befindet.

DEFAULY IO BYTE SETTINGS

Der nächste Programmpunkt, 'Default IO byte settings', ist wohl hauptsächlich etwas für Hardware-Bastler, die ihre neuesten Geräte ins Betriebssystem einbinden wollen. So könnte z. B. der Parallel-Drukkerausgang auf eine – noch zu bauende – serielle Schnittstelle umgelegt werden. Die Treibersoftware für zwei serielle Anschlüsse befindet sich übrigens schon im Betriebssystem'-ROM! Nähere Einzelheiten dazu finden Sie im 'Floppy-Buch' von Data-Becker.
Die weiteren Programmteile sind sehr speziellen Anwendungen vorbehalten, darum seien sie nur kurz erwähnt:
- Slow/Fast mode
- BIOS messages enabled/disabled
- Clear/Preserve initial command buffer on keyboard input

Weiter geht es mit einigen Möglichkeiten zur Veränderung der Betriebsdaten der Floppy-Station. Haben Sie z. B. eine 5 1/4-Zoll- oder 8-Zoll-Diskettenstation angeschlossen, kann es sein, daß z. B. die Stepper-Rate nicht mit dem Amstrad-Standard übereinstimmt. Im einzelnen können Sie folgende Einstellungen vornehmen:
- **MOTOR ON DELAY:** Dies ist die Zeit, die nach dem Einschalten des Floppy-Motors vom Computer gewartet wird, damit dieser seine Nenndrehzahl erreichen kann. Normalerweise ist der Wert 1 sec, Sie können aber in 1/50-Sekunden-Schritten alles einstellen.
- **MOTOR OFF DELAY:** Ist analog dazu die Zeit beim Abschalten des Laufwerk-Motors, der Normalwert sind 5 Sekunden.
- **DEFAULT STEPPING RATE:** Dies ist die Zeit, die der Lese- und Schreibkopf des Laufwerks zum Sprung von einer Spur zur nächsten benötigt. Haben Sie als Zweit-Laufwerk etwa ein besonders langsames erwischt, können Sie es damit doch noch zum Laufen bringen.

Die nächsten Programmteile sind wieder für Bastler, denn damit kann man die Daten zweier serieller Schnittstellen einstellen. Das Betriebssystem ist auf die Verwendung zweier Z80-SIO-Bausteine ausgelegt.
Nach all diesen Anpassungen wollen Sie natürlich Ihre CP/M-Diskette mit diesen Änderungen versehen. Geben Sie also bei der Frage 'Do you want to update your system disc?' ein Y für Yes ein. Als nächstes können Sie CP/M neu starten, um die Änderungen auszutesten.
Sind Sie mit dieser CP/M-Version zufrieden, sollten Sie mit BOOTGEN.COM diesen Boot-Sektor auf alle Ihre Disketten kopieren. ***Martin Kotulla***

Ergänzung des Beitrages von Martin Kotulla „Wohin mit Maschinenprogrammen auf dem CPC-464?“, aus Tips u. Tricks von CPU (Nr. 9)“

Winfried Huslik, Buchauto des Titels „CPC-464 inside out“, ließ uns folgende Ergänzung (teilweise Korrektur) zu diesem Beitrag zukommen, den wir an dieser Stelle veröffentlichen.
Es ist tatsächlich möglich, Maschinenprogramme unterhalb von HIMEM abzulegen, man muß nur den „richtigen“ Trick anwenden. Um das folgende zu verstehen, muß man sich den Zustand des Speichers nach dem Einschalten klar machen:

AC00	Beginn der BASIC-Daten
AB80:	SYMBOL-TABELLE der 16 obersten Zeichen (SYMBOL AFTER 240)
	Bereich für BASIC-String-Variable

Wird nun durch ein Kommando HIMEN oder ein OPENOUT oder OPENIN das Ende der Stringbereiche nach unten verschoben, werden die letzten 16 Zeichen „eingefroren“, es ist also nicht mehr möglich, ein SYMBOL AFTER durchzuführen, da keine Möglichkeit besteht, den Bereich zu verändern.
Beginnt man dagegen mit dem Kommando SYMBOL AFTER 256, so verschwindet die USER SYMBOL TABLE völlig. Man kann dann durch MEMORY beliebig viel Speicherplatz für eigene Zwecke reservieren. Werden selbst definierte Zeichen benötigt (für den deutschen Zeichensatz immer), kann anschließend ohne weiteres mit SYMBOL AFTER eine neue USER SYMBOL TABLE angelegt werden.
In diesem Zusammenhang soll-

te man berücksichtigen, daß wohl kaum ein Programm ohne I/O auskommt. Es empfiehlt sich deshalb, den Pufferbereich gleich mit zu reservieren. Die komplette Initialisierung sieht dann so aus:

```
10 SYMBOL AFTER 256
20 OPENOUT „DUMMY“ (richtige Anführung einsetzen)
30 MEMORY (himem-1)-Platz.für.Maschinenprogramme
40 CLOSEOUT
50 SYMBOL AFTER 32
```

Der Speicher sieht dann so aus:

AC00:	Beginn der BASIC-Daten
	4k Puffer für Datei-Ein- und Ausgabe
	Platz für Maschinen-Programme nach Bedarf
HIMEN	USER SYMBOL TABLE
	String-Bereich für BASIC usw.

Dieses Verfahren hat einen Nachteil, aber viele Vorteile. Der Nachteil: Auf der Diskette wird tatsächlich eine Datei namens „DUMMY“ angelegt und bei wiederholtem Programmlauf auch noch eine „DUMMY.BAK“. Das kostet zwei Datei-Einträge und zwei Blöcke auf der Diskette.

Vorteil 1: Sollte Bedarf für weitere Maschinenprogramme erwachsen, so kann mit SYMBOL AFTER 256:MEMORY HIMEM-(neuer.Platz+1):SYMBOL AFTER ... weiterer Platz belegt werden.

Vorteil 2: wurde bereits vielfach veröffentlicht, nämlich braucht bei OPEN-Kommandos kein GRABAGE COLLECT durchgeführt werden; da der I/O-Buffer fest zugeordnet ist.

Zu den Programmen in der Funktionszeichentabelle:

Genauso könnte man die Programme im Bereich der SOUND ENVELOPES unterbringen. Dort oder im Bereich der TEXT STREAM PARAMETER dürfen aber keine ENVELOPES benutzt werden und deshalb kann man hier nur mit Window #0 arbeiten.

Natürlich gibt es etliche Nieschen für kleine Maschinenprogramme im RAM-Bereich des CPC-464. Hierfür empfehle ich mein Buch „CPC-464 – inside out“.

Zu den Programmen in Stringvariablen

Dies dürfte in der Tat eine sehr interessante und flexible Möglichkeit sein sich eine Programmbibliothek aufzubauen, doch das in der CPU Nr. 9 abgedruckte Beispiel beinhaltet einige Probleme.

Mit CALL @a$ ruft man keineswegs ein Maschinenprogramm im String A$ auf, sondern man springt unkontrolliert in den Speicher! Warum?

Das Basic-Kommando CALL erwartet als Argument eine Adresse, bestehend aus zwei Bytes. @A$ liefert zwar einen Zwei-Byte-Wert, doch handelt es sich dabei nicht um die Adresse von A$, sondern um die Adresse des Stringdescriptors von A$. Dieser wiederum besteht aus drei Bytes, nämlich der Länge und (jetzt kommts) der String Adresse. Um ein Programm in einem String aufzurufen, muß man sich doch eines Zwischenschritts bedienen:

```
a%=@a$ : CALL PEEK(a%+1)+PEEK(a%+2)*256
```

Winfried Huslik

Tips und Tricks für den VC-20

Save und Load

A. SAVE für Maschinenspracheprogramme im Kassettenpuffer.

Hierbei handelt es sich in erster Linie um Maschinenspracheroutinen, die vom Programm erzeugt und im Kassettenpuffer abgelegt werden; dabei muß man darauf achten, daß die MS-Routine erst ab Zelle 849 beginnt. Im Direktmodus eingeben:

```
A$="":FOR N = 849 to 1019:
A$= A$ + CHR$(PEEK(N)):
NEXT
```

Wenn READY erscheint: SAVE"Programmname[mit genau 16 Zeichen]+A$

Wenn der Name kürzer als 16 Zeichen ist, füllt man mit SPACE den Rest auf.

B. Wer den Kassettenpuffer benutzen will, kann MSRoutinen bis zu einer Länge von 88 Bytes in den Zellen 678 bis 767 ablegen, da diese vom Betriebssystem nicht benutzt werden.

C. Wer den Kassettenpuffer für MSRoutinen benutzt, sollte wie unter A. angegeben vorgehen und erst ab Zeile 849 mit der Routine beginnen, damit diese Routine zusammen mit dem Tapeheader geladen wird und nicht jedesmal von neuem erzeugt werden muß. Danach kann man die Programmzeilen für die Routine löschen und so Speicherplatz im Hautpspeicher sparen. Außerdem bietet dieses Verfahren einen Schutz gegen gedruckte Raubkopien des Listings.

D. SAVE für einen bestimmten Speicherbereich.

Häufig möchte man den Bildschirminhalt abspeichern. Mit folgenden Schritten im Direktmodus ist das kein Problem:

1. POKE Startadresse des Speicherbereiches in 193(LSB) und 194(MSB)
2. POKE Endadresse in 174(LSB) und 175(MSB)
3. POKE 186, Gerätenummer (1=Kassette,8=Diskette)POKE185,3
4. POKE 147, 0 für LOAD oder 1 für VERIFY
5. Save mit SYS63109 (es wird KEINE Meldung 'SAVING' ausgegeben)
6. Load mit SYS62795 (es wird KEINE Meldung 'FOUND' ausgegeben)

Bei 5. und 6. nach Eingabe von RUN sofort die entsprechende Taste(n) [PLAY bzw. PLAY+RECORD] drücken.

BEISPIELPROGRAMM:

```
 10 PRINT
    CHR$(147)CHR$(5):
    POKE36879,8:X=0
 20 Beginn des zu speichernden Bildschirminhalts
    (bis99)
100 GOSUB500
110 PRINT "BILDSCHIRM
    LADEN?"
120 GET A$: IF A$="N"
    THEN END
130 IF NOT A$ = "J" THEN 120
140 GOSUB600:X=0
150 GOTO150:REM Bildschirm 'einfrieren'
500 GOSUB700
510 IF PEEK(37148)=254
    GOTO510
520 SYS63109:RETURN
600 PRINT CHR$(147):
    GOSUB700
610 SYS62795: RETURN
```



```
700 POKE193,0:
    POKE194,PEEK(648)
710 POKE174,0:
    POKE175,
    PEEK(648)+2
720 POKE186,1: POKE185,3
730 POKE147,X: RETURN
```

E. Mit folgendem Trick kann man die Tastatur in Grafik- oder Textmodus schalten und verriegeln:

Für Textmodus:

```
A$=CHR$(14)+CHR$(8)+Programmname:SAVE A$
```

Für Grafikmodus:

```
A$=CHR$(142)+CHR$(8)+
Programmname:SAVE A$
```

Soll die Tastatur NICHT verriegelt werden, läßt man + CHR$(8) weg.

F. Wenn man mehrere Kopien eines Programmes abspeichern will, kann man sich einiges an Tipperei einsparen, indem man mit den Farben arbeitet: zuerst die Bildschirm löschen und mit SAVE die erste Kopie abspeichern; nach Erscheinen von READY Cursor auf SAVE setzen und mit CTRL+Zahl eine neue Cursorfarbe wählen und dann RETURN drücken.

Bei Zeilengruppen, die in einem Programm mehrmals unverändert, oder nur geringfügig verändert erscheinen, kann man auf diese Art und Weise auch übersichtlichere Abtrennung der Programmsegmente auf dem Bildschirm bzw. auf einem farbfähigen Drucker erreichen.

G. Bei zweiteiligen Programmen, die aus Anleitungsprogramm und eigentlichem Programm bestehen, kann man durch folgende Zeile das Anleitungsprogramm vom Programm aus loeschen und das eigentliche Programm laden:

```
POKE198,1:POKE631,131:NEW
```

H. Variablen retten.

Variablen kann man vor Verlust durch das Ausschalten retten, indem man sie in einem Datafile abspeichert. Um z. B. Variable A$ zu retten, ist folgende Programmierung möglich:

```
1000 OPEN1,1,1, "filename“
1010 PRINT#1,A$
1020 CLOSE1
```

Will man, im Normalfall, mehrere Variablen speichern, muß jede Variable einer Zeile wie 1010 zugewiesen bekommen.

Um die Variablen wieder zu laden:

```
1100 OPEN1,1,0,"filename“
1110 INPUT#1,A$
1120 CLOSE1
```

Auf diese Weise kann ein unterbrochener Programmlauf zu einem späteren Zeitpunkt an der Abbruchstelle fortgesetzt werden: die Verwendung dieser Methode bei selbstprogrammierten Spielen, um den Spielstand zu retten, ist nur eine mögliche Anwendung.

I. Software-Scutz: noch ein Vorschlag.

Vor dem Save, in Direktmodus eingeben:

1. POKE849,PEEK(4097)POKE850,PEEK(4098)
2. POKE4097,0:POKE4098,0

SAVE mit einem Programmnamen von genau 16 Stellen, ggfls. mit Leertaste auf 16 Stellen auffüllen.

Nach dem Laden eines so 'geschützten' Programmes:

```
POKE4097,PEEK(849):POKE4098,PEEK(850)
```

Es können selbstverständlich ganz andere Zellen gePEEKed werden, bloß sollte man nicht zu viele verschiedene Kombinationen verwenden, sonst kann man u.U. seine eigenen Programme nicht mehr zum Laufen bringen.

J. Noch einmal: Kassettenpuffer

Wer auf seine Kassettenpuffer beim Nachladen oder Neuladen von Programmen nicht verzichten möchte, hat die Möglichkeit, den Kassettenpuffer zu verlegen: zum Beispiel, mit POKE 178,0: POKE 179,PEEK(648) kann man den Kassettenpuffer auf den Anfang des BildRAMs verschieden. Das Bild ist zwar nicht besonders, aber die MS-Routine wird nicht überschrieben. Wer Wert auf ein ordentliches Bild legt, kann sich ein anderes 200-Byte-langes Stück RAM aussuchen und die Adresse in die Speicherzellen 178 und 179 – im üblichen LO/HI-Byte Format einPOKEn.

Nützliche Formeln: Tips u. Tricks für den VC-20

A. Umwandlung von ASCII-Werte in BildschirmPOKEs.

```
DEF FN(A) = A-161-33*(A
<255)-64*(A<192)-32*(A-
160)+32*(A<96)-64*(A<64)
```

B. Umwandlung von BildschirmPOKEs in ASCII-Werte.

```
B=B+128*(B >127): B=B-
64*(B <32 OR B >95)-32*(B
>63 AND B <96)
```

C. Logarithmus Basis 10.

```
DEF FN L(X) = LOG(X)/
LOG(10)
```

D. Potenz Basis 10.

```
DEF FN E(X) = EXP(X*-
LOG(10))
```

E. MOD Funktion: der Rest bei einer Division. DEF FN MOD(X) = INT((A/B-INT(A/B))*B+0.5)

Eine Variante lautet:

```
DEF FN MOD(X) = INT(A-
B*INT(A/B))
```

F. Rundung.

```
DEF FN R(X) = INT(X/DS+
0.5)*DS
```

rundet die Zahl X mit dem in DS (=Dezimalstellen) angegebenen Wert: z.B. DS=0.01:Z=FNR(X):PRINT Z rundet die Zahl auf zwei Dezimalstellen

z.B. DS=10 :Z=FNR(X): PRINT Z rundet die Zahl auf den nächsten Zehnerwert.

Tips und Tricks

Pokes, Peeks und Calls für den Apple II

Peeks:

X=PEEK (37)	vertikale Cursorposition ermitteln
X=PEEK (36)	horizontale Cursorposition ermitteln
X=PEEK (-16286)	PDL (1) – Taster lesen
X=PEEK (-16287)	PDL (0) – Taster lesen
X=PEEK (222)	ERROR-Code nach ON ERRor lesen
X=PEEK (218) + PEEK (2119) *256	Zeilennummer, bei der ein Fehler aufgetreten ist (bei ON ERROr-Anweisung)
X=PEEK (-16384)	Tastatur lesen (X enthält Tastatur-Code)
X=PEEK (175)+PEEK(176) *256)-(Peek(103)+PEEK (104)*256)	Länge des Basic-Programms im Speicher
X=PEEK (103) + PEEK (104) * 256	Beginn des Basic-Programms im Speicher
X=PEEK (175) + PEEK (176) * 256	Ende des Basic-Programms im Speicher

X=PEEK (105) + PEEK (106) * 256	Beginn der Variablen eines Basic-Programms
X=PEEK (107) + PEEK (108) * 256	Ende der Variablen eines Basic-Programms
X=PEEK (107) + PEEK (108) * 256	Beginn der Arrays eines Basic-Programms
X=PEEK (109) + PEEK (110) * 256	Ende der Arrays eines Basic-Programms
X=PEEK (115) + PEEK (116) * 256	Ende der Strings eines Basic-Programms
X=PEEK (234)	Collision Counter (Anzahl der Punkte, die ein Shape mit einem vorher gesetzten Shape gemeinsam hat)
X=PEEK (-16336)	Ausgabe eines 'Klick' vom Lautsprecher
X=PEEK (-16352)	Ausgabe eines 'Klick' am Cassetten-Ausgang

Pokes:

POKE 214,255	sämtliche Basic-Befehle wirken wie RUN
POKE 50,127	FLASH auch in INTBASIC
POKE 50,255	NORMAL auch in INTBASIC
POKE 50,63	INVERSE auch in INTBASIC
POKE -16304,0	auf Graphik-Mode umstellen
POKE -16298,0	schaltet auf Lowres-Graphik um (Graphik mit niedriger Bildschirmauflösung)
POKE -16297,0	schaltet auf Hires-Graphik um (Graphik mit hoher Bildschirmauflösung)
POKE -16303,0	auf Text umschalten
POKE 37,X	vertikale Cursorposition setzen
POKE 36,X	horizontale Cursorposition setzen
POKE 216,0	ON ERRor - Flag löschen
POKE -16368,0	Tastatur-Flagge löschen
POKE 32,X	linker Rand des Bildschirmfensters
POKE 33,X	Breite des Bildschirmfensters
POKE 34,X	oberer Rand des Bildschirmfensters
POKE 35,X	unterer Rand des Bildschirmfensters
POKE 231,X	SCALE (Ändern des Shape - Vergrößerungsfaktors)
POKE 249,X	ROT (Ändern des Shape - Rotationsfaktors)

Calls:

CALL -336	löschen des Basic-Programms (NEW)
CALL -958	Bildschirm ab Cursor löschen
CALL -868	Bildschirmzeile ab Cursor löschen
CALL -936	Bildschirm löschen
CALL -912	Bildschirm eine Zeile hochrollen
CALL -715	Einzeltaste lesen
CALL -756	auf Tastendruck warten
CALL -211	Ausgabe von 'ERR' und Ctrl-G
CALL -676	Ausgabe von Ctrl-G und warten auf <RETURN>
CALL -198	Ctrl-G ausgeben
CALL -1321	Ausgabe der Registerinhalte
CALL -673	auf <RETURN> warten
CALL -151	Einsprungadresse des Monitors
CALL 1002	Reaktivierung von DOS, dann zurück ins Basic

Einfachdatenübertragung im Telefonnetz

Dateneingabe mit dem Telefonapparat

Wie bei einem Telefongespräch wird die gewünschte Zentrale angerufen. Die dort installierte Endeinrichtung meldet sich, z. B. mit einer rechnergesteuerten Sprachausgabeeinrichtung. Die Dateneingabe erfolgt über die Datentastatur des Telefonapparates, eine Kontrolle der eingegebenen Daten kann ggf. durch Wiederholen der Eingabe über die Sprachausgabeeinrichtung erfolgen. Anwendungsbeispiele sind Auskunftssysteme, Bestellsysteme usw.
Die Übertragungsgeschwindigkeit beträgt maximal 10 Zeichen pro Sekunde; es steht ein Vorrat von 16 verschiedenen Zeichen zur Verfügung.

Als Zentralstation wird ein Modem D10P-Z verwendet, an den eine private Datenendeinrichtung mit Parallel-Schnittstelle nach CCITT-Empfehlung V. 19 angeschlossen wird. Sofern die Datenendeinrichtung keine Sprachausgabe besitzt, kann auch ein vom Modem erzeugter 420 Hz-Ton, durch die Datenendeinrichtung gesteuert, als Empfangsquittung ausgesendet werden.
Als Außenstation bietet die Deutsche Bundespost folgende Telefonapparate an:

- Telefonapparat 71, ein Telefon für Mehrfrequenzwahl. Einsatz bei Hauptanschlüssen an elektronischen Vermittlungsstellen (z. B. EWS) und in elektronischen Nebenstellenanlagen.
- Telefonapparat 78, ein Telefon, das mit einer 12-teiligen Wähltastatur für das Impulswahlverfahren und einer 16-teiligen Tastatur zur Dateneingabe ausgestattet ist.

BTX im Kommen! - Das Kommunikationssystem der Zukunft?

- Telefonapparat 80, ein Telefon mit einem 16-teiligen Tastenblock, in dem die Funktionen Impulswahl und Dateneingabe in einer Tastatur vereinigt sind.

Außerdem können für die Dateneingabe private Zusatzeinrichtungen mit akustischer Ankopplung an die Sprechmuschel des Handapparates verwendet werden. In diesem Fall ist die Benutzung eines beliebigen Telefons, auch Münzfernsprecher, möglich.

* * * * * * *

Programmiersprachen im Vergleich

Für viele Heimcomputer und für fast alle Personalcomputer gibt es eine Unmenge von Programmiersprachen. Um die Qual der Wahl zu erleichtern, werden an dieser Stelle die wichtigsten vorgestellt.

BASIC: Beginners All Purpose Symbolic Instruction Code; wurde 1965 in den USA entwickelt. Diese Sprache ist in fast allen Heimcomputern fest im ROM integriert. BASIC ist leicht zu erlernen, aber nicht ausreichend strukturiert; d. h. man findet sich in fremden Programmen nur schwer zurecht. Aufgrund der unzählbar vielen BASIC-Dialekte sind die Programme wenig portabel. Ein Atari Programm läuft beispielsweise nicht auf einem Schneider-Rechner.
LOGO: Ebenfalls in den 60er Jahren in Massachusetts, USA, entwickelt. Zwar war LOGO ursprünglich als Programmiersprache speziell für Kinder vorgesehen, doch über diesen Status ist man längst hinaus. Nicht nur die Turtle-Graphik spricht für LOGO, sondern auch die rekursive Programmiertechnik (Nachvollziehen der menschlichen Denkstruktur). Der neue Atari ST Personalcomputer soll sogar serienmäßig mit LOGO ausgestattet sein.
C: C ist eine noch relativ junge Programmiersprache, die unter anderem zur Programmierung des 16-/32- bit Betriebssystems UNIX verwendet wurde. C hat außerordentlich große Zukunftschancen und wird stark an Bedeutung gewinnen. Diese Sprache ist maschinennah und damit ein idealer Assembler-Ersatz.C - Programme sind sehr portabel.
FORTH: Auch Forth steht der Assembler-Sprache sehr nahe und ist damit, wie auch C, wirklich schnell. Die Befehle können durch die Definition eigener Programm-Module ständig erweitert werden. Forth wurde von einem Engländer namens Moore entwickelt und bedeutet soviel wie „die vierte Generation".
PASCAL: PASCAL wurde in Zürich Ende der 60er Jahre entwickelt. Es erfordert eine sehr strukturierte Programmierung, Pascal-Programme sind sehr übersichtlich. Allerdings wird dies teuer erkauft: Die Programmierung in Pascal gestaltet sich oft sehr umständlich. Es schien eine Zeit lang, daß Pascal sich einen zweiten Platz hinter BASIC sichern würde. Aber nun steht fest, daß LOGO weitaus bessere Chancen hat. Und die kommende 16-/32-bit Generation, z. B. die 68000-Computer Sinclair QL, Atari 520 ST, Apple MacIntosh, etc. werden sich wohl ebenfalls eher mit C anfreunden.
FORTRAN: FORmula TRANslator; entwickelt in den 50er Jahren. FORTRAN hat einen technisch-wissenschaftlichen Charakter und eignet sich sehr zur Programmierung von mathematischen Problemen.
COMAL: Common Algorithmic Language; eine Programmiersprache, die BASIC ähnelt. In Comal wird die strukturierte Programmierung unterstützt, wie dies auch in Pascal der Fall ist. COMAL soll wie auch BASIC recht leicht zu erlernen sein. Die Verbreitung von Comal ist jedoch noch ziemlich gering.
ASSEMBLER: Assembler gehört nicht zu den sogenannten „höheren" Programmiersprachen wie BASIC, LOGO, Pascal, etc. Assembler ist eine Maschinensprache mit der Verwendung von Mnemonics. Mnemonics sind Abkürzungen, die anstatt des Zahlencodes verwendet werden. Ein solches „Quell-Programm" wird beim Start des Assemblers in den „Objektcode", das eigentliche Maschinenprogramm, umgesetzt. Assembler ist nicht leicht zu erlernen und setzt viel Geduld voraus.
COBOL: Common Business Oriented Language; in den 60er Jahren vom US-Verteidigungsministerium entwickelte Programmiersprache. Cobol ist eine sehr leistungsstarke Sprache für kaufmännische Probleme, die für die Bewältigung großer Datenmengen ausgelegt ist. Cobol ist leicht auf fremde Computer zu übertragen. Cobol - Programme sind sehr übersichtlich.
ELAN: Elan wurde in den 70er Jahren in den Universitäten in Berlin und Bielefeld entwickelt. Abgesehen von Großrechnern wurde Elan auch für den Z 80 - Mikroprozessor implementiert. Elan gehört zu den stark strukturierten Sprachen.
PL 1: Programming Language 1; von IBM Ende der 60er Jahre entwickelt. PL 1 soll sich für die Lösung sowohl mathematisch-wissenschaftlicher als auch kaufmännischer Probleme eignen. PL 1 - Programme lassen sich in Gruppen und Blöcke einteilen, wodurch der Programmierer dafür sorgen kann, daß auch nur der wirklich benötigte Speicherplatz im RAM reserviert wird.
LISP: Zu Beginn der 60er Jahre in den UA entwickelte Programmiersprache. Mit LISP wurde ein Versuch auf dem Wege zur „artificial intelligence", kurz AI, gestartet. AI, auch KI genannt, bedeutet soviel wie „künstliche Intelligenz". Die Stärke von LISP liegt in der Rekursion. Daher besteht auch eine gewisse Verwandschaft zu LOGO.

Außer den hier vorgestellten 12 Sprachen gibt es noch viele mehr: ADA, Pearl, MODULA 2, ALGOL, PILOT, APS, APL, etc. Als Fazit könnte folgendes gesagt werden: Wer Spiele programmieren möchte, ist mit dem schnellen, aber nicht leichtem Assembler am besten beraten. Auch Forth und C bieten sich für schnelle Spiele an. Für kaufmännische Problemstellungen ist nach wie vor COBOL zu empfehlen. Technisch-wissenschaftliche sowie mathematische Probleme lassen sich mit Fortran, Pascal, C und LOGO lösen. Als Programmiersprache für den Einsteiger als auch für den Profi sind die neueren BASIC - Versionen trotz aller Kritik weiterhin gut geeignet. Allerdings sollte die Verwendung von Prozeduren und lokalen Variablen zur Programmstrukturierung Voraussetzung sein. Wenn Sie Ihre Programme ausreichend in REM - Zeilen dokumentieren und chaotische GOTOs vermeiden, können Sie auch in BASIC einen hervorragenden Programmierstil erreichen!

Patrick Schmitz

ACHTUNG!

Wir suchen noch weitere gute

Tips & Tricks

Meldet Euch. Ein Honorar winkt!

Tips und Tricks...

für die wichtigsten

Beliebte und bekannte Sortierverfahren

Computersysteme

In diesem Beitrag sollen die bekanntesten Sortierverfahren erklärt und verglichen werden. Eine der Hauptaufgaben der Datenverarbeitungsanlagen besteht in der Sortierung von großen Datenmengen. Auch bei Home- und Personalcomputern werden vielfach Sortieralgorithmen eingesetzt, um beispielsweise die Adressendatei in alphabetischer Reihenfolge ausgeben zu können.

Mittlerweile gibt es eine große Zahl von Sortieralgorithmen, die sich alle mehr oder weniger unterscheiden. Die hier abgedruckten Programme sind für den Apple II geschrieben, es ist allerdings problemlos möglich, diese auf andere Rechner zu übertragen, da nur die Minimal-Befehle von Basic verwendet wurden.

Hierbei handelt es sich um ein relativ einfaches Verfahren, die Methode der LINEAREN AUSWAHL, können Sie Listing 1 entnehmen.

Grundlage für die kurze Routine ist das unsortierte Array A, das sortierte Ergebnis wird in dem ebenso großen Array B abgelegt. Die Anzahl der Elemente (wichtig für die Dimensionierung von A und B) ist durch N festgelegt.

Listing 1 (Lineare Auswahl):

```
100 FOR I=1 TO N
110 ZEIGER = I
120 FOR J=2 TO N
130 IF A(J) < A(ZEIGER)
    THEN ZEIGER=J
140 NEXT J
150 B(I)=A(ZEIGER)
160 A(ZEIGER)=9.9E37 :
    REM GROESSTE
    MASCHINENZAHL
170 NEXT I
180 REM ZAHLEN SIND
    NUN SORTIERT
```

Wie funktioniert nun dieser recht einfache Algorithmus?

Das Programm bestimmt zunächst aus der unsortierten Liste das kleinste Element und stellt es an die erste Position der Ergebnisliste. Dazu wird die Originalliste mit der zweiten For-Next Schleife linear durchsucht (daher auch der Name LINEARE AUSWAHL).

Die Position, in der in der unsortierten Liste das kleinste Element stand, wird (in Zeile 160) mit der größten Maschinenzahl des Rechners gefüllt. Warum dies auf diese Weise geschieht, wird im folgenden deutlich.

Ist nun das erste Element in der Ergebnisliste abgelegt, so beginnt der Algorithmus erneut mit dem Suchen nach dem kleinsten Element in der unsortierten Liste. Nun wird das ursprünglich zweitkleinste Element gefunden, da das kleinste ja mit der größten Maschinenzahl überschrieben wurde. Hier liegt also der Grund für Zeile 160.

Das ursprünglich zweitkleinste Element wird in der Ergebnisliste an die zweite Position gelegt.

Dieses System wird N-mal ausgeführt (äußere For-Next-Schleife), dann sind alle Zahlen in aufsteigender Reihenfolge sortiert.

Dieses Sortierverfahren ist (auf Grund seines relativ simplen Systems) nicht sonderlich schnell. Für 100 Zufallszahlen benötigt der Algorithmus auf dem Apple II etwa 81 Sekunden. Außerdem benötigt er für die Daten recht viel Speicherplatz. Für N Zahlen werden 2N Speicherplätze gebraucht. Schon deshalb ist es für große Datenmengen ungeeignet.

Das zweite Verfahren, das vorgestellt wird, ist der Klassiker unter den Sortiergorithmen: der BUBBLE-SORT.

Listing 2 zeigt den Sortieralgorithmus in einer Version für den Apple II, er ist jedoch ebenfalls problemlos übertragbar:

(Listing 2: Bubblesort)

```
100 FOR I=1 TO N-1
110 FOR J=1 TO N-1
120 IF A(J)>A(J+1) THEN
    A(0)=A(J+1):A(J+1)=
    A(J):A(J)=A(0)
130 NEXT J
140 NEXT I
150 REM ZAHLEN SIND
    NUN SORTIERT
```

Wie beim Verfahren der linearen Auswahl sind die unsortierten Zahlen in dem Array A gespeichert. N gibt wieder die Anzahl der Daten an.

Wie arbeitet der Bubblesort? Verdeutlichen wir die Arbeitsweisen an einem Beispiel:

Die Zahlen 5, 11, 3 und 7 sind zu sortieren. Der Tabelle können Sie die jeweiligen Variablenwerte zu dem Zeitpunkt entnehmen, der durch die Werte von I und J angegeben wird.

I	J	A(1)	A(2)	A(3)	A(4)	Bemerkung
0	0	5	11	3	7	Anfangsposition
1	1	5	11	3	7	5<11 also kein Tausch
1	2	5	3	11	7	11<3 also Tausch
1	3	5	3	7	11	11>7 also Tausch
2	1	3	5	7	11	5>3 also Tausch
2	2	3	5	7	11	5<7 also kein Tausch

Da die richtige Reihenfolge schon hergestellt ist, wird von nun an nicht mehr getauscht, die Schleifen werden nutzlos durchlaufen.

Beim Bubblesort wird eine Zahl stets mit ihrem Nachfolger verglichen. Ist sie größer, so werden die beiden vertauscht. Auf diese Art und Weise stehen die beiden Zahlen schon „etwas richtiger". Nun wird nach diesem Verfahren weiterhin verglichen. Somit „wandert" eine große Zahl stets weiter nach oben (in den Bereich großer Zahlen), eine kleine Zahl hingegen bewegt sich auf den Anfang des Arrays zu.

Wie wir schon an unserem Beispiel gesehen haben, arbeitet auch der Bubblesort nicht sonderlich effektiv. Zwar wird nur noch die Hälfte des Speicherplatzes als bei der linearen Auswahl benötigt, doch laufen die Schleifen des Bubblesorts häufig nutzlos weiter, da die Zahlen (besonders bei vorsortierten Daten) oft frühzeitig richtig geordnet sind.

Auf dem Apple II benötigt der Bubblesort zum Sortieren von 100 Zufallszahlen etwa 122 Sekunden und ist somit nochmal etwas langsamer als der „einfache" Algorithmus der LINEAREN AUSWAHL! Das dritte u. letzte Sortierverfahren trägt den Namen SHUTTLE-SORT. Das Listing 3 zeigt den Algorithmus in der Apple-Form:

(Listing 3: Shuttle-Sort)

```
100 FOR I=1 TO N-1
110 IF A(I) <
    A(I+1) THEN 180
120 A(0) = A(I+1)
130 C=I
140 A(C+1)=A(C)
150 C=C-1
160 IF A(0) <
    A(C) THEN 140
170 A(C+1) = A(0)
180 NEXT I
190 REM ZAHLEN SIND
    NUN SORTIERT
```

Auch diesen Algorithmus wollen wir genau erklären.

Beim Shuttle-Sort werden die Daten vom Beginn an geprüft, ob sie in bezug auf ihren Nachfolger in der Liste an der richtigen Stelle stehen. Ist dies der Fall, so wird dieser Nachfolger entsprechend mit seinem Nachfolger verglichen. Dieses Verfahren setzt sich solange fort, bis zwei Elemente falsch stehen. In diesem Fall wird das an der falschen Position stehende Element in A(0) zwischengespeichert. Anschließend werden alle vorigen Elemente aufgerückt, die größer als das in A(0) abgelegte Element sind. Anschließend wird das Element A(0) an seine richtige Position gesetzt, die ja nun frei ist.

Ist die For-Next-Schleife durchgelaufen, so sind N-1 Daten an die richtige Position gebracht worden. Damit muß auch die letzte Zahl richtig angeordnet sein.

Der Shuttle-Sort sortiert 100 Zahlen in etwas 45 Sekunden. Damit hat sich die Zeit der LINEAREN AUSWAHL fast halbiert.

„Logoleien" mit dem Computer

BOOLEANs, sicher haben Sie schon viel darüber gehört. In PASCAL, LOGO tun sie brav ihre Dienste, haben Sie gedacht!! Im SCHNEIDER-BASIC gibt es sie auch: sie haben die Form -1 für TRUE und 0 für FALSE.

Wahrscheinlich haben Sie auch das schon gewußt, aber haben Sie auch schon gewußt, daß Sie mit ihnen auch außerhalb der IF-Anweisung arbeiten können und solche sogar selbst erzeugen können? NEIN?! Dann will ich Ihnen mit diesen Tricks zeigen wie sie BOOLEANs handhaben:

1. Erzeugung von BOOLEANs außerhalb von IF, NOT, OR, AND und WHILE Anweisungen (Sie können die zuerst erwähnten Routinen aber auch verwenden wenn Ihnen das lieber ist):

Beispiel einer LET Anweisung bzw. einer Variablenzuweisung:

```
10 LET c=(a=b)
```

In dem gezeigten Fall würde c bei Gleichheit der Vergleichsvariablen a und b den Wert -1 für TRUE beinhalten. Das Gleiche gilt für sämtliche anderen nichtarithmetischen Operatoren.

2. Verwendung des IF und WHILE Statements ohne Vergleich mit Hilfe der Verwendung von BOOLEANs:

Da die Vergleiche bei IF und WHILE im Endeffekt ohnehin BOOLEANs erzeugen wäre es doch einfacher das BOOLEAN zuerst zu erzeugen, dann in die Variable abzuspeichern und dann x IF Anweisungen einfach statt dem umständlichen 'a=c THEN' ein dezentes Boolean hinzusetzen, was viele IF Anweisungen doch um einiges verkürzt.

3. BOOLEAN als Ersatz von umständlichen NOT, OR, AND's die obendrein auch noch sehr begrenzt verknüpfbar sind (Vorteil: mathematische Schreibweise von logischen Verknüpfungen wird angenähert und Verquickung von arithmetischen Verknüpfungen und logischen Verknüpfungen wird möglich):

Stellen Sie 'Logikklammer'-Ausdrücke neben arithmetische Operationen und verbinden Sie sie arithmetisch oder logisch miteinander.

Sie können auch quasi unendliche logische Verknüpfungen herstellen.

Sicher finden Sie auch noch eine spezielle Anwendung der von mir beschriebenen Möglichkeiten.

Nun viel Spaß an Ihrem Computer bei den „Logoleien" mit den so trickreichen Booleans!

Ein Tip für die User des C 16- bzw. des C 116-Computers

Wenn sich der Computer aufhängt, nicht gleich ausschalten oder Reset-Taste drücken, sondern: Reset-Taste niederdrükken und gedrückt halten, Run-Stop-Taste niederdrücken und gedrückt halten, Reset-Taste loslassen, Run-Stop-Taste loslassen. Hierdurch gelangt man in den Monitor-Bereich, und mit „X" und Return wieder zum Basic.

Sinn und Zweck dieses Tips: Das Programm bleibt erhalten, man kann sofort listen und starten etc.

Das Ganze funktioniert dann, wenn sich der Computer in einer sogenannten Programmschleife befindet.

SOFT-TEAM-BERLIN
Software und Zubehör für
C-64 Atari Schneider
Spectrum MSX
Über 2000 Programme immer auf Lager.
Ständig die neuesten Programme aus England und den USA.
Gratiskatalog anfordern:
Anton Peter & Partner, Kamminerstraße 9
1000 Berlin 10, Telefon 453 27 11

HALLO!
„Gewerbliche": Dieser „RAUM" ist zu verkaufen! Bitte Media-Unterlagen anfordern!

Wir haben zwei Super-Chancen, die man sich nicht entgehen lassen sollte!

Dies ist was für scharfe Rechner!

CPU Bestellkarte-Softwareservice

Alle im Heft abgedruckten Programme können als zusätzlicher Service über den Verlag bezogen werden.
(Ausland nur gegen Vorkasse)

Die Zustellung erfolgt: gegen **Vorkasse** ☐ oder Inland per **Nachnahme** ☐ + Versandkosten

innerhalb von 1 Woche

Entnehmen Sie bitte aus unseren Preislisten die notwendigen Angaben für Ihre Bestellung:

Bitte liefern Sie mir

☐ Cassette für _____ (System) ☐ Anzahl

☐ Diskette für _____ (System) ☐ Anzahl

zum Preis von gesamt _____ DM

Name/Vorname: _____ Straße, Nr.: _____

PLZ/Ort: _____ Datum, Unterschrift

Abonnements:

wer jetzt in's Computerleben einsteigen will, sollte nicht überlegen: ein Abonnement ist immer eine lohnende Sache!

Ausschneiden und einsenden an:

Verlagsunion
Friedrich-Bergius-Straße 20
Postfach 57 07
6200 Wiesbaden

Bestellgutschein

Ja, ich möchte ins Computerleben einsteigen und bestelle deshalb:

☐ Jahresabonnement **CPU** (6 Hefte) 60,– DM
☐ Jahresabonnement **HC und Computronic** (12 Hefte) 60,– DM
☐ Jahresabonnement **Kombi: HC, Computronic und CPU** (18 Hefte) 85,– DM
(Ausland: siehe Impressum)

Name/Vorname _____

Straße, Nr. _____ PLZ, Ort _____

Ich wünsche folgende Zahlungsweise:
☐ Bargeldlos durch Bankeinzug _____ Bankleitzahl

Geldinstitut _____ Konto-Nr. _____

☐ gegen Rechnung

Datum, Unterschrift _____

Abonnements-Kündigungen:
6 Wochen vor Ablauf des Jahresabonnements.
Außerdem kann ich diese Bestellung innerhalb von 10 Tagen beim Verlag widerrufen!
Zur Wahrung der Frist genügt die rechtzeitige Absendung des Widerrufs!

Datum, Unterschrift _____

Hier sind die ersten Gewinner unseres Tips & Tricks (CPU)-Autoren-Wettbewerbs:

In unserem ersten Heft „TIPS & TRICKS" von CPU hatten wir zu einem Autoren-Wettbewerb aufgerufen, bei dem insgesamt 600 Mark zu „ergattern" waren. Eine wahre Flut an Einsendungen mußte geordnet, geprüft, getestet und bewertet werden. Das „CPU"-Team hat es sich nicht leicht gemacht, die besten Autoren-Beiträge zu honorieren. Da es jedoch nun mal nur drei Gewinner gibt, wurden folgende Preisträger ermittelt:

Das beste USER-Programm – DIGITAL SOUND EDITOR (für den Apple – schrieb nach Meinung der Redaktion **Roland Graf** aus Strassberg 1. Ihm wurden die redlich verdienten 300 Mark „zugedacht".
Der Programmbeitrag DISC-HILFE (CPC 464) von **Joachim Glaubrecht** aus Pulheim wurde von uns als bestes UTILITY-Programm ausgezeichnet. Lohn für diese Arbeit: 200 Mark.
„Last but not least" gingen 100 Mark an **David Twigg-Flesner** aus Gronau-Epe, der einen hervorragenden Beitrag lieferte. Titel: ALLGEMEINE HILFEN FÜR DEN VC 20.

In der nächsten Ausgabe sind Sie an der Reihe, „Liebe Leser!" Dann nämlich fällt das schwierige Amt des Juroren Ihnen zu. Die Leserschaft fungiert als Jury zur Bewertung der Programmbeiträge selbst. Wer sich als „Jury-Mitglied" beteiligt, hat zwei Vorteile: Zum ersten die Chance, einen Datenträger zu gewinnen (bitte geben Sie Ihr Computer-System bei der „Bewerbung" mit an!), und zum anderen die Möglichkeit, zur kostenlosen Kleinanzeige in der nächsten „TIPS & TRICKS von CPU". Den entsprechenden Coupon finden Sie auf dieser Seite!

Also, schreiten Sie zur Tat! Teilen Sie uns schriftlich mit, wer Ihrer Meinung nach die besten USER-, UTILITY- und TIPS und TRICKS-Beiträge geliefert hat. ***(Siegfried Görk)***

Mitmachen!
Mitwirken!
Mitgewinnen!

Schreiben Sie an:
TRONIC-Verlag
Redaktion TIPS & TRICKS
z. Hd. Herrn Görk
Postfach 41
3444 Wehretal 1

Kleinanzeigen

Liebe Leser,

wollen Sie eine Kleinanzeige aufgeben? Suchen Sie einen gebrauchten Home-Computer, Software, Hardware, Kontakte zu Computer-Clubs etc.?

Dann nutzen Sie jetzt den Kleinanzeigenservice der „CPU": Die „CPU" ist ein kompetentes Anwender-Magazin.

Sie erreichen durch uns eine Vielzahl von Lesern im In- und Ausland.

Und so wird's gemacht:

Kleinanzeigen-Coupon ausfüllen (alle Zeilen können ausgefüllt werden) und **einsenden an**

Tronic-Verlag
Postfach 41
3444 Wehretal 1

Also, schnell ausfüllen und einwerfen in den nächsten Postkasten. Dann ist mit Sicherheit Ihre Kleinanzeige dabei.

- CPU - Basar

Kleinanzeigen

- CPU - Basar

Bitte veröffentlichen Sie ab nächstmöglicher Ausgabe unter der Rubrik:

☐ SOFTWARE ☐ HARDWARE ☐ VERSCHIEDENES

Mein Computersystem: _____

folgenden Text:

Bitte jeweils 28 Buchstaben pro Zeile! Bitte Absender nicht vergessen!

Preis für Kleinanzeigen: 10 DM je Kleinanzeige
(keine gewerbliche (Nutzung) ☐ Scheck ☐ Bar
☐ kostenlose Kleinanzeige für unsere Juroren (siehe S. 91)!

Name/Vorname: _____

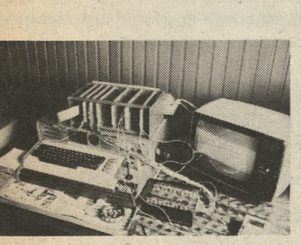
Straße, Nr.: _____ PLZ/Ort: _____

Datum, Unterschrift _____

Ich versichere hiermit, daß die angebotenen Programme/Beiträge frei von Rechten Dritter sind!

Universelles Bussystem für VC-20 und C-64 (Teil 2:)

Nach Veröffentlichung des 1. Teils unserer Hardware-Serie erhielt die Redaktion viele positive Zuschriften und überdies zahlreiche Bestellungen für die abgedruckten Baupläne in Originalgröße.
Nutzen Sie also unseren Service: Wir liefern Ihnen gegen einen (noch nicht feststehenden) Unkostenbeitrag die gesamten Bauanleitungen, Schaltpläne, Platinenpläne usw., in der Originalgröße, die Sie benötigen, um Ihren VC-20 bzw. C-64 als universellen Steuerungscomputer einsetzen zu können. Falls Sie die betriebsbereiten Platinen erwerben möchten oder bei speziellen Fragen, die diese „Serie" betreffen, wenden Sie sich bitte direkt an den Autor:
Helmut Thöne – Telefon 0 56 76 / 13 76 – Di. u. Do. 18-19 Uhr.



Buskarte für C-64

Beim Belichten der Platine wird die Vorlage spiegelbildlich aufgelegt. Zum Bohren der Platine sollten nach Möglichkeit für die IC's 0,8 mm-Bohrer verwendet werden. Bei den restlichen Löchern kann man auch einen 1 mm-Bohrer benutzen.
Zuerst werden die Drahtbrükken wie auf dem Bestückungsplan ersichtlich ist, montiert, danach erfolgt das Einlöten der IC-Sockel. Für das E-Prom wird ein 28-poliger Sockel verwendet, falls man dabei auf variable Programme in Maschinensprache verzichtet, kann die Fassung weggelassen werden. Nun kann man die 31-polige Steckerleiste für die Verbindung zum Bus anlöten und daran anschließend die beiden Kondensatoren C1 + C2 (bei C2 auf Polung achten!!). Dann wird der Widerstand R1 montiert und die Verbindung zum C-64 hergestellt.
Für die Verbindung gibt es mehrere Möglichkeiten, z.B. kann eine 31-polige Steckerleiste und das Flachbandkabel an der 31-poligen Buchsenleiste angelötet werden.
Bei der Buskarte wurde die Steckerleiste weggelassen und stattdessen das Flachbandkabel direkt an die mit (x) gekennzeichneten Bohrungen angelötet.
Bei der Verbindung zwischen Buskarte und Platinenstecker (wird hinten in den 44-poligen Expansionsport des C-64 eingesteckt) ist darauf zu achten, daß alle gleichnamigen Punkte miteinander über das Flachbandkabel verbunden werden.
Das bedeutet; die oberste Bohrung mit der Bezeichnung D-Clock auf der Buskarte wird mit der gleichen Bohrung auf dem Platinenstecker (wird ebenfalls mit D-Clock bezeichnet) verbunden, usw..
Wie zu erkennen ist, sind die Anschlüsse auf dem Platinenstecker in der gleichen Reihenfolge wie auf der Buskarte gehalten. Zum Abschluß werden die IC's in die Fassungen eingesteckt, dabei ist unbedingt auf die richtige Polung zu achten.

Arbeitsweise der Buskarte

Zunächst das Signal SØ2. Dieses Signal bildet den Systemtakt des Rechners, durch diesen werden die gesamten Schreib- und Lesezugriffe gesteuert. Die Taktfrequenz beträgt ca. 985 KHz. Bei dem im Bussystem verwendeten Interfacebaustein VIA 6522 muß dieses Signal geringfügig verzögert werden. Man hätte auch die gleichen VIA vom Typ 6526, wie im C-64 verwenden können, aber diese IC's sind teuer und nur sehr schwer zu erwerben.
Die erwähnte Verzögerung geschieht folgendermaßen: Im Mittelpunkt steht das IC4 vom Typ 74LS74. Wir gehen davon aus, daß SØ2 gerade Low-Pegel hat und die Leitung SØ2 mit dem Reset (Pin 1) und dem Daten-Eingang (Pin 2) verbunden ist. Wird der Reset-Kontakt Low, so wird der Ausgang (Pin 5) ebenfalls Low. Jetzt geht die Leitung SØ2 auf High und im Reset und im Dateneingang liegt dann High-Pegel. Allerdings werden die Daten erst von Pin 2 nach dem Ausgang (Pin 5) übertragen, wenn Pin 1 und Pin 2 High und der Takt, Eingang (Pin 3) High wird. Jetzt fehlt nur noch der Takteingang. Das Signal D-Clock schwingt mit der 8-fachen Frequenz von SØ2, dadurch wird erst mit der nächsten ansteigenden Flanke von D-Clock am Takteingang das Signal von D nach Q übertragen und das Signal SØ2 vom C-64 geringfügig verzögert auf den Bus übertragen. Versucher haben ergeben, daß eine geringfügige Verzögerung von D-Clock durch die R/C-Kombination von R1 und C1 erfolgt, so daß das SØ2-Signal soweit verzögert wird, daß die VIA 6522 auf den einzelnen Steckkarten problemlos arbeiten kann.
IC3 ist der Adressdecoder für die 8 Steckplätze (BU2-BU9). Die Ausgänge Pin 7,9-15 sind die ENABLE für die einzelnen Steckplätze. An Pin 1-3 liegen die Adressleitungen A4-A6. Diese drei Eingänge sind für die Auswahl der acht Ausgänge zuständig, Voraussetzung ist jedoch, daß Pin 6 High, Pin 4 und

5 Low-Pegel hat. Pin 5 ist fest mit Ø Volt verbunden. Wenn in den Bus geschrieben wird oder aus dem Bus gelesen wird, geht IØ1 auf Low, wird invertiert und erscheint als High am Pin 6, gleichzeitig muß die Adressleitung A7 Low haben. Damit ist das IC freigegeben. Liegt jetzt an Adressleitung A4-6, Pin 1-3 Low-Pegel, so befindet sich ENØ Pin 15 auf Low-, die anderen auf High-Pegel. Soll anschließend der EN7, Pin 7 auf Low gehen, müssen die Adressleitungen A4-6 auf High liegen. A4 hat für den Adressdecoder die Wertigkeit 1, A5 die Wertigkeit 2 und A6 die Wertigkeit 4. Man kann somit in drei Leitungen 8 einzelne Ausgänge schalten. Der angesteuerte Ausgang hat Low-, die anderen High-Pegel. Die Steckplätze BU2-BU9 auf dem Bus sind jeweils mit einem Ausgang von IC3 verbunden. Da immer nur einer der 8 Ausgänge von IC3 Low-Pegel hat, ist jeweils nur einer der 8 Steckplätze freigegeben.
Wer noch keine Erfahrung mit Binärzahlen besitzt, sollte sich durch die entsprechende Literatur informieren.
Beispiel:
Jeder der 16 Adressleitungen ist ein Wert zugeordnet.

1	2	4	8	16	32	64	128
A0	A1	A2	A3	A4	A5	A6	A7

...usw.
Geht man davon aus, daß im Steckplatz 3 des Busses (BU3) eine Eingangskarte steckt, werden anschließend die Eingänge von Port B abgefragt. Der Eingang PB4 soll EIN, PB0-3 und PB5-7 aus (Ø) sein.

Das Signal IØ1 wird im C-64 schon soweit adressiert, daß dieses Signal immer dann einen Low-Pegel besitzt, wenn der Adressbereich von DEOO bis DEFF hexadezimal und 56832 bis 57087 dezimal angesprochen wird. Diesen Adressbereich benötigt man für die Steckkarten.

Das Signal IØ1 wird durch N1 invertiert, d.h. ist IØ1 Low, ist am Ausgang von N1, Pin 11 High-Pegel und umgekehrt. Dieses Signal wird einem Eingang von N2 zugeführt. Der zweite Eingang von N2 kommt von SØ2.
Nur wenn IØ1 Low (IØ1 ist Low, wenn die Adressen auf dem Bus angesprochen werden) und SØ2 High ist, wird der Ausgang des Gatters N2 Pin 8 Low. Dieses Signal wird dem IC2 vom Typ 74LS245 am Pin 19 angelegt, dadurch erreicht man die Freigabe des birektionalen Bustreibers. Ist Pin 19 Low, so werden über diese IC die Daten vom C-64 zum Bus oder vom Bus zum C-64 übertragen. Diese Richtungsänderung geschieht mit dem Signal CR/W. Das Signal wird über die beiden Gatter N3 und N4 zweimal umgedreht, so daß am Ausgang von N4 Pin 6, der gleiche Pegel wie am Eingang von N3 steht. Soll jetzt in den Bus geschrieben werden (Poke), so wird das Signal CR/W Low. Der Schreib/Lese -Eingang von IC2 Pin 1 wird Low. Dann werden die verschiedenen Pegel von den Datenleitungen (Pin 11-18) auf die Kontakte 2-9 (Datenbus) übertragen. Dieses funktioniert jedoch nur, wenn gleichzeitig Pin 19 von IC2 auf Low-Pegel liegt. Liegt der Freigabeeingang auf High-Pegel, so ist das IC in beiden Richtungen hochohmig.
Die Adressleitungen A0-A3 werden von IC1 gepuffert und auf den Bus gegeben. Diese vier Adressleitungen sind für die Auswahl der 16 Adressen je Steckplatz zuständig.
Die beiden Leitungen RESET und IRQ werden über die Buskarte nur geschleift.
Die Wertigkeigt für PB0 = 7 sind aus

PB0 = 1	Aus
PB1 = 2	Aus
PB2 = 4	Aus
PB3 = 8	Aus
PB4 = 16	Ein
PB5 = 32	Aus
PB6 = 64	Aus
PB7 = 128	Aus

Da nur PB4 + sein soll, ergibt sich eine Zahl von 16, die in Port B steht.
Aus Tabelle 2 (siehe CPU 9) ist ersichtlich, daß die Basisadresse von BU3 56848 ist. Diese Adresse bildet auch gleichzeitig die Ein-Ausgabe Adresse von Port B.
Beispiel:
Im Programm steht: A = Peek (56848), dadurch geht IØ1 auf Low. Die Basisadresse von Ø1 ist 56832. Da Peek 56848 eingegeben wurde, wird jetzt davon 56832 abgezogen, daraus ergibt sich die Wertigkeit 16. Für die Wertigkeit der 16 Adressleitungen steht nun die Adressleitung A4, gleichzeitig ist I01 Low und A4 High.
Nun zu IC3: Die Leitung A4 ist High (Pin 1), A5 und A6 sind Low. Das ergibt für den Adressdecoder eine Wertigkeit von 1. Da die Adressleitung A7 (Pin 4) Low, Pin 5 fest mit 0 Volt verbunden und Pin 6 High ist (I01 ist über N1 invertiert), ist das IC freigegeben (siehe oben). Weil die Wertigkeit 1 beträgt, wird EN 1, Pin 14 Low. Somit wird der Steckplatz 3 auf dem Bus BU3 über die Drahtbrücke A angesteuert. Jetzt ist das Interface IC auf der Eingangskarte angewählt und die Schreib/Leseleitung CR/W wird High, dadurch kann IC2 über Pin 1 zur Datenübertragung in den C-64 und gleichzeitig in das Interface IC auf der Steckkarte auf Datenausgabe umgeschaltet werden. Erst wenn die Leitung SØ2 auf High geht, gelangen die beiden Eingänge des Gatters N2 ebenfalls in den High-Pegel. Der Ausgang des Gatters 2 wird ebenso wie Pin 19 des IC2 auf Low gesetzt, dadurch ist dann IC2 freigegeben. Gleichzeitig wird auch das Interface IC auf der Steckkarte mit dem Signal SØ2 "High" freigegeben. Dieses IC gibt jetzt den Wert 16 aus Post B auf den Bus. Die Daten (16) erscheinen an den Pin 2-9 auf der Buskarte. Da das IC durch Pin 1 in Richtung des C-64 geschaltet ist, werden die Werte an den Pin 11 bis 18 gelegt, diese sind über Flachbandleitungen mit dem Expansionsport (D0-D7) verbunden. Der Prozessor liest den Datenbus im Rechner und weist anschließend den Wert 16 (D0-D3 ist Low, D4 ist High und D5-D7 ist Low) der Variablen A zu.

In kurzen Beispielen wird die Programmierung der einzelnen Bausteine erläutert, u. a. die Handhabung von Ein- und Ausgabe, Richtungsregister, Timer lund 2, Schieberegister, Steuerregister und Interrupt-Register. Dies sind grundlegende Voraussetzungen um für die Steckkarten eigene Programme schreiben zu können. Zwar werden zu jeder Steckkarte die entsprechenden Programme vorgestellt, aber universell für jeden Anwendungsfall ist das System nur, wenn in Verbindung mit den verschiedenen Steckkarten eine eigene anwendungsorientierte Programmerstellung erfolgen kann.

Programmierung der Buskarte

Auf der Buskarte befindet sich ein Platz für ein E-Prom. Wer z. B. ein festes Programm für eine Steckkarte geschrieben hat, kann dieses Programm, sofern es in Maschinensprache erstellt ist, in ein E-Prom brennen und jeder Zeit mit Sys aufrufen. Es können E-Proms 2732 = 4 k Byte oder 2764 = 8 k Byte Speicher verwendet werden. Diese Typen sind zur Zeit auf dem Markt am einfachsten erhältlich. Wenn die Buskarte mit E-Prom betrieben werden soll, muß auf dem Platinenstecker eine Brükke gesteckt werden (Auf dem Bestückungsplan des Platinensteckers als Brücke gekennzeichnet). Dadurch wird die Leitung Game an 0V gelegt. Jetzt ist das normalerweise am Adressbereich 8000-9FFF liegende interne RAM abgeschaltet. Gleichzeitig wird das Signal ROM L als Chip Select Signal für das E-Prom benutzt.
Für den Betrieb mit E-Prom muß auf der Buskarte noch die Brücke 1 hergestellt werden. Damit wird das Signal A12 mit dem Pin Output ENABLE am E-Prom verbunden.
Damit ein E-Prom Daten ausgeben kann, muß erst einmal die Adresse auf die Adressleitung gelegt (A0-12) und die beiden Pin Output ENABLE und Chip Select gegen Masse gezogen werden. Jetzt werden die Daten, die in den Adressen liegen über die Datenleitung D0-D7 ausgegeben:

Wird ein E-Prom 2732 (4KByte) auf der Buskarte verwendet (Achtung, richtig einstekken!!!!) so liegt der Adressbereich von 8000-8FFF. Ab der Adresse 9000-9FFF liegt zwar auch die Leitung ROM L auf 0V aber die Adressleitung A12 geht auf High, so daß der Eingang "Output ENABLE" ebenfalls auf High geht. Das E-Prom kann nun keine Daten mehr ausgeben.
Bei der Verwendung des E-Prom 2764 (8 KByte) wird der gesammte Speicherbereich von 8000-9FFF abgedeckt.
Diese max. 8 K Byte dürften für ein eigenes Steuerprogramm ausreichen.
In diesem Bereich liegt auch der Autostart. Er funktioniert folgendermaßen. Nach dem Einschalten ooder bei einem RESET (RESET TASTER) fragt der Computer die Speicherzellen S 8004-8008 ab und vergleicht, ob die einzelnen Speicherzellen folgenden Inhalt aufweisen:

```
S 8000 09 ┐
S 8001 80 ┘── Start Adresse zeigt auf S 8009
S 8004 C3 ┐
S 8005 C2 │   C
S 8006 CD │── B ── Test auf Autostart
S 8007 38 │   M
S 8008 30 ┘   8
              0

S 8009 LDX 00
S 800B STX D016     Bildschirm erstellen
       JSR FF84     CIA's Initialisieren
       JSR FF87     Page 0-3 löschen RAM Test
       JSR FF8A     I/O einschalten
       JSR FF81
       ...  ...
       ...  ...
       ...  ...
```

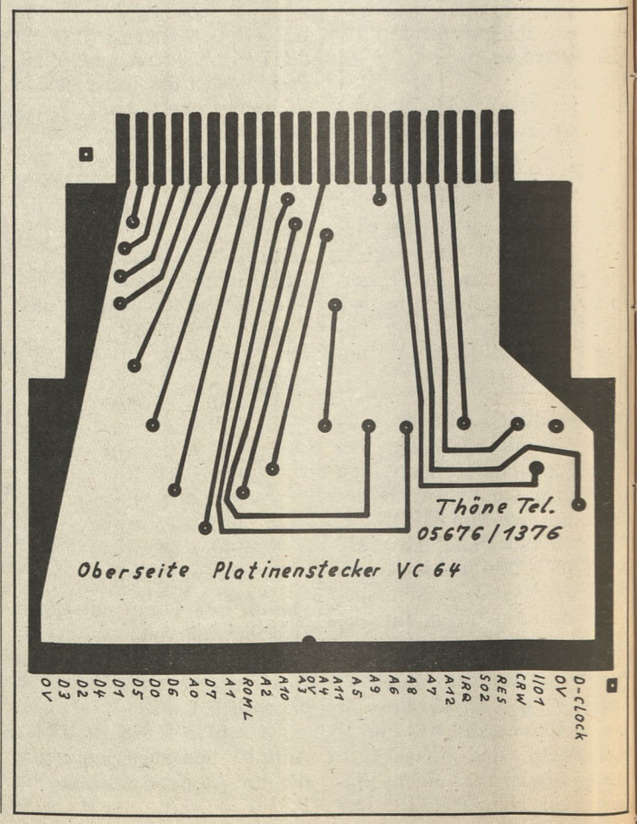
Ab hier eigenes Programm.
Wird kein Autostart erwünscht, so wird das Programm gleich ab 8000 geschrieben. Beim Einschalten prüft der Computer die 5 Speicherzellen, ob der Code CBM80 enthalten ist. Besitzt eine der Speicherzellen einen anderen Inhalt, bricht der Computer den Test ab und meldet sich in Basic, anschließend kann das Programm mit Sys gestartet werden.
Wenn beispielsweise die Zentralheizung mit dem Computer gesteuert wird, so käme bei jedem Stromausfall das Programm zum Stoppen. Um auch sicherzugehen, daß sich der Computer bei starken Störungen nicht aufhängt, kann man (über eine Ausgangsleitung der Ausgangskarte) durch das Programm Impulse erzeugen, die ein Monoflop, das nachtriggerbar ist, ansteuert. Wenn sich jetzt der Rechner verabschiedet, erfolgt keine Nachtriggerung und nach dem Ablaufen der Mono-Flopzeit wird aus dem Ausgang des ersten Monoflop ein zweites getriggert, das einen Low Impuls auf die RESET-Leitung des Rechners übergibt, der sich durch den Autostart wieder im Programm befindet.

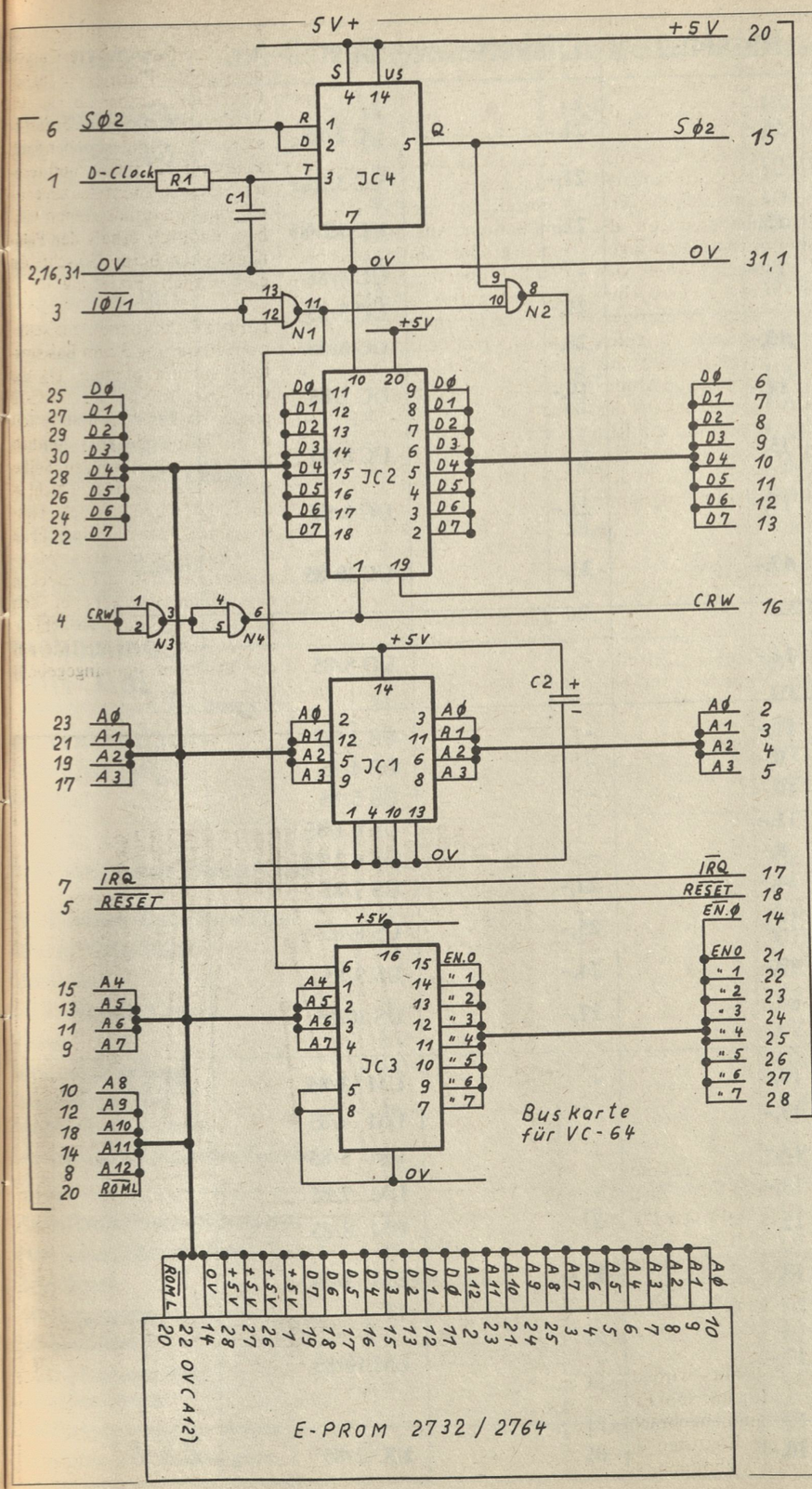
Der Platinenstecker

Man benötigt eine doppelseitige Platine, entweder ein beschichtete oder man muß diese mit Fotolack besprühen. Zuerst wird die Unterseite des Platinenlayouts (Seitenverkehrt - spiegelbildlich -, die Schrift darf nicht lesbar sein) auf eine Seite der Platine gelegt und mit einer Glasplatte beschwert, dann belichtet.
Nach dem Belichten wird die Platine kurzzeitig mit dem Entwickler zusammengebracht. Sobald sich die Konturen des Layouts im Lack abbilden, sofort die Platine aus dem Entwicklerbad nehmen und mit klarem Wasser spülen. Jetzt die Platine auf eine weiche Unterlage legen, damit die andere Platinenseite mit dem Fotolack nicht verkratzt wird und die beiden Lötungen, die mit x gekennzeichnet sind, mit ca. 1 mm bohren. Anschließend mit einem größeren Bohrer, die beiden Bohrungen auf beiden Seiten ein wenig absenken und dann die Platine mit der Oberseite des Platinenlayouts so bedecken, daß diese nicht seitenverkehrt (Schrift muß lesbar sein) auf der Platine liegt. Danach müssen noch die beiden Bohrungen auf der Platine (in den Lötaugen mit x gekennzeichnet) übereinander zu liegen kommen. Wenn das Layout sich in der richtigen Position befindet, dieses vorsichtig mit einer Glasplatte beschweren und dann kann belichtet, entwickelt und geätzt werden.

Beim Sägen des Platinensteckers benutzt man für den C-64 am besten die äußere Umrandung an der Platine, die auf der Oberseite sichtbar ist. An der Vorderseite, die in die Federleiste des C-64 eingeschoben wird, sollte auf der linken und rechten Seite ca. 2 mm des abgeätzten Basismaterials stehen bleiben, dadurch erhält der Platinenstecker beim Einschieben eine seitliche Fixierung.

Beim VC-20 können zur seitlichen Fixierung 3 mm Basismaterial stehen bleiben. Da die Gehäuseöffnung des VC-20 größer ist, kann der hintere Teil des Platinensteckers seitlich breiter gehalten werden. Sollte sich der Platinenstecker nicht in die Federleiste einschieben lassen, muß man mit einer Feile auf beiden Seiten etwas Material entfernen. Beim Einstecken des Platinensteckers in den Rechner ist darauf zu achten, daß dessen Oberseite (wie auf den Platinenfolien angegeben) nach oben zeigt.





Buskarte zum C-64

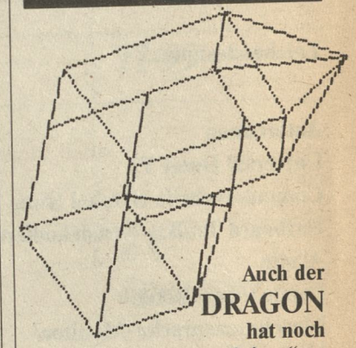
Gesucht werden bei:

Kontaktadresse:
c/o Bernd Becker
Wittener Str. 69 c
4354 Datteln

Schneider CPC 464 Computer Club Datteln 1985 sucht Mitglieder
Wir wollen:
1. Erfahrungsaustausch
2. Programmverkauf (Originale)
3. Programmtausch (Selbstgeschriebene)
4. Club-Info (4-8 wöchentlich)
5. Treffen zwecks gemütlichen Beisammensein und Klönen.

Bitte meldet Euch mit Freiumschlag an obige Adresse!!

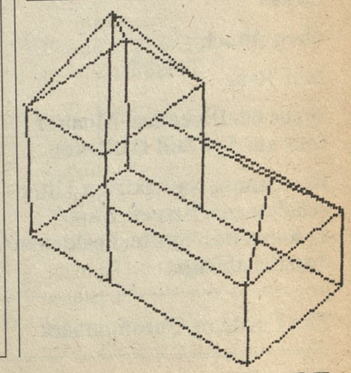
VORSCHAU



Auch der DRAGON hat noch „einiges" zu bieten: Im nächsten Heft stellen wir ein wahres TOP-SUPER - ANWENDER - PROGRAMM vor!

Es heißt:

3-D-Rotation



Programm	Kassette	Diskette	Bestell-Nr.	System
				Commodore 64
Multi-Key/S-Tool/ Interrupt-Programm	16,–	21,–	UC 2/85	
Spritehilfe/Diskloader/Directory	–	21,–	UC 3.1/85	
Tape-Directory/Asmon/Data-Generator	15,–	21,–	UC 3.2/85	
Fast-Load	10,–	–	UC 4/85	
Diskmonitor/Zeichensatz	–	20,–	UC 5/85	
Reassembler/Maskengenerator	15,–	20,–	UC 6/85	
Del 64/Treiberprogramm für 1526/MPS 802 Decelerator	14,–	19,–	UC 7/85	
Supertape-Directory/Renew/ Controllport-Abfrage	15,–	–	UC 8/85	
Video-Utilitys/ Multi-Basic V3/Bildeditor	17,–	23,–	UC 9/85	
Better Basic/Autonumber/ Hardcopy/Terminalprogramm	17,–	23,–	UC 10/85	
				Commodore C 16/C 116
Zeichendesigner	14,–	–	UO 8/85	
				Schneider CPC 464
Autorennen	11,–	–	US 3/85	
Universal Datei V1	9,–	–	US 4/85	
Computerschrift/Symbol Swap	10,–	–	US 5/85	
Keyboard Toolkit/Farbdekodierung	11,–	–	US 6.1/85	
Aladin	9,–	–	US 6.2/85	
Painter/Box-Befehl	14,–	24,–	US 7/85	
Maschinensprache-Monitor/ Disk-Hilfe	15,–	25,–	US 8/85	
Basic/Maschinenkit	12,–	24,–	US 9/85	
Back-up/Directory/ Ellipse/Disk	–	27,–	US 10/85	
				MSX
Zeichensatzkonverter	10,–	–	UM 3/85	
Pipeline	13,–	–	UM 4/85	
Zirkus	13,–	–	UM 5/85	
Alien Attack	13,–	–	UM 7/85	
Anti-Rom	12,–	–	UM 8/85	
Typer 60/Disketten-Monitor (nur auf 5¼ Zoll Diskette)	19,–	25,–	UM 9/85	
Berechnung von aktiven Filter-schaltungen/Screen-Save/ Z-Konverter/Sprite-Positionen/ Joystickabfrage	17,–	–	UM 10/85	
				ZX-81
The Castle of Questionmark	10,–	–	UX 2/85	

Bestellen Sie telefonisch: ☎CPU-Software-Service: 0 56 51 / 4 06 93 oder 4 06 43

Programm	Kassette	Diskette	Bestell-Nr.	System
				VC-20
Sprites/Space-Battle	13,–	19,–	UV 2/85	
Grafik-Painter	10,–	–	UV 5/85	
Tic-Clock	9,–	–	UV 6/85	
Decelerator	9,–	–	UV 7/85	
Joypaint	12,–	–	UV 8/85	
Disassembler/Old-Programm	10,–	–	UV 9/85	
Directory/Farb-JRQ	10,–	–	UV 10/85	
				Apple
Asteroids	–	10,–	UA 2/85	
Dateiverwaltung	–	18,–	UA 3/85	
Digiter/Reset-Saver	–	18,–	UA 4/85	
Assembler-Monitor	–	17,–	UA 5/85	
Disk Check IC/Dos Change	–	18,–	UA 6/85	
Hello	–	17,–	UA 7/85	
As-Editor	–	18,–	UA 8/85	
Digital-Sound-Editor/ Basic-Befehle	–	18,–	UA 9/85	
Big Spender	–	19,–	UA 10/85	
				TI-99
Olympics	11,–	–	UT 2/85	
Funktionsanalyse/Hardcopy	11,–	–	UT 3/85	
Invaded by the Empire	11,–	–	UT 4/85	
Zeichendefinition	14,–	–	UT 8/85	
Superfile	11,–	–	UT 9/85	
				ZX-Spectrum
Panik Sam/Colour Change/Clock	13,–	–	UZ 2/85	
MC-Routinen/Data-Base/Pixel	11,–	–	UZ 3/85	
Reversi/UDG-Editor	15,–	–	UZ 4/85	
Star Basic	13,–	–	UZ 5/85	
Fill-Programm/Copy-it	14,–	–	UZ 6/85	
Quassi	13,–	–	UZ 7/85	
Intersound/Balken	14,–	–	UZ 8/85	
Trace-Single Step/ Variable List/X-Basic	16,–	–	UZ 9/85	
Disassembler/Ohmsches Gesetz/	17,–		UZ 10/85	
				Atari
Mauerflucht/Lunar-Simulation	13,–	19,–	UI 2/85	
Ram-Tester/Cherry Harry	13,–	–	UI 3/85	
Crazy Egon	13,–	19,–	UI 5/85	
Monitorprogramm	11,–	–	UI 6/85	
Zeicheneditor	–	19,–	UI 7/85	
Kopfzeilengenerator/ Lohnabrechnung	–	22,–	UI 8/85	
Grafikausgabe/Error-Helper	–	18,–	UI 9/85	
Translator/Zeichendesigner	14,–	20,–	UI 10/85	

Hier ist immer 'was zu holen: der CPU-Bazar - bunt, interessant

Verkäufe

C64 Willst Du umtauschen oder kaufen, Software / Kopier-Programme, Spiele u.a. / Schreibe an / Liste beilegen oder 2,-- DM in BM / Ierzy Zabierek, vl. Rolnicza 93, Ø5-Ø92 Lomianki.

TI 99/4 Ext.-Spiel nur für Erwachsene – bitte Alter ang. zum absoluten Preishit von nur DM 10,– im Umschlag an T. Karbach, Remscheider Str. 18, 5650 Solingen 1.

Vereinsbestandsführung oder Sportwettkampf + Ergebnisliste Cartridge 54 DM. Cass. 38 DM, Info gegen Freiumschlag 80 Pf, R. Wendlandt, Beckerskamp 26, 4300 Essen 14.

Neuheit für TI99/41. Endlich Schluß m. 28 Zeichen Alpha Datek Y.1.0. Ein Datei Prg. mit 40 Zeichen pro Zeile, echte Kleinb. mit Unterlänge, und und und . . . erforderl. EX-B, 32K&DSK Info By U. Brüsseler: 02166/ 53457, Wetschewell 125 D, 4050 MG 3, ab 20 Uhr.

CPC 464 Superhardcopy für NLQ 401, Größe 27,7 x 16,8 cm, also etwa Bildschirmgröße, Cass. 20 DM, Disk. 30 DM in Scheinen od. Scheck bei Norbert Möndjen, Lindenstr. 77, 7071 Spraitbach.

Funkfernschreib-Hard- und Software (RTTY) RX + TX bis 200 Baud, ASCII und Baudot, Tel.: 030/7063879 ab 17 Uhr.

Verk. Top TI 99/4A + Cass. Rec. + P-Box, kompl. m. int. Contr. Disk., 32 K, RS 232 + Soyst. + Ex Bas + Ed. Assem. + Sprachsynt. + 8 Mod., VB = 2800 DM, M. Sandmann, Im Branke 5, 4630 Bochum, Tel. 382596.

CPC 464!! Auf jeder Kass./Disk 12 gute Spiele. Keine Raubkopien. Nr. 1 = 20 DM, Nr. 2 = 30 DM. Alle 24 Programme nur 40 DM (Disk. + 10 DM). Scheine/Scheck o. Nachnahme. H.J. Behrendt, Neustadt 3, 3550 Marburg.

Hallo TI-Freaks: Nicht Verkaufen, sondern Ausbauen. 32 kRAM - Speichererweiterung sehr günstig! Info gegen Rückporto bei: H. Ayasse, Schöckeng. 5, 7253 Renningen 2.

ZX-81 + Cassettenrekorder + 16 K + Buch für sage und schreibe 200,-- DM! Bitte melden bei: Marcus Dzionsko, Tel. 0202/597828, August-Mittelsten-Scheidstr. 4, 56 Wut 2.

Genie 16 B – 128 K – deutsch inclusive 2 Basicversionen + Compiler + Turbopascal wegen Systemwechsel für 4750 DM. Tel. 0711/421599 nur von 13.00 bis 14.00 Uhr.

ATI Rex Soft präsentiert: Decathlon, das Supersportspiel jetzt auch für den TI!! Fordert noch heute den Katalog an!! Für 2 DM bei: D. Peter, Hulfteggstr. 31, CH-8400 Winterthur.

CPC 464!!! 12 gute Anwender Progr., keine Raubkopien, Kass. 30 DM, Disk. 40 DM. Scheine/Scheck o. Nachnahme. H. Behrendt, Marrensberg 2, 2390 Flensburg.

Verkaufe VC-20 (Defekt) – 75 DM, 32 W Erw. – 100 DM, Datasette – 50 DM, ca. 300 Superprogramme (auch einzeln!). Liste für 80 Pf von: P.v.D. Most, Neustr. 32, 4432 Gronau, Tel.: 0256220555.

Verk, VC-20, mit 12 Modulen, 13 K Superexpander, 18 K Erw., 8 Büchern u. 6 Spielkassetten für 600 DM, Stefan Sommer, Josephskirchstr. 13, 6000 Ffm. 50, Tel. 926991

ZX-Spectrum, Top-Software, Tausch oder Verkauf, sendet Eure Liste oder 2,-- DM in BM an Maria Buszkowska, u.l. Meksykanska 3 m62, Ø3-951 Warszawa, Polen.

Verkaufe und **Tausche** gute Software für den **VC 20.** Liste gegen 1 DM in Briefmarken. Sven Faulhaber, Bösinghäuser Weg 3, 3401 Waake.

Ankäufe

Suche Ex-Basic für den **TI-99/4A** Angebot an: Markus Pesth, Dientzenhoferweg 4, 8402 Neutraubling.

Spectravideo 328: Suche Kontakt-Programm und Erfahrungsaustausch: Kassette oder Disk. Kjell Aaberg, Munkarpsv./5, S 24300 Höör, Schweden.

Suche gutes Lottoprogramm, Heinz Ziegler, Wilhelmsruher-Damm 133, 1000 Berlin 26. Tel. 030/4153566.

*** **CPC 464:** Suche Kassetten und Erfahrungsaustausch: Jede volle Kassette an mich kommt mit Info-File + Programm voll zurück. P. Anlehla, Brenner Str. 42, 8600 Bamberg.

Clubs

SVI/MSX-Computerclub: Zielgruppe sind Besitzer von SVI-Computern (SVI-318/-328/-728) und Bondwell-Computern (12/14/16). Aktivitäten: Clubzeitschrift (3monatlich) – Erfahrungsaustausch – Software – Börse mit Austausch von selbstgeschriebenen Programmen – Kontakte zu anderen User-Clubs – Telefonischer Feuerwehrdienst für akute Soft- und Hardware - Schwierigkeiten unter den Tel.-Nr. 0209/205243, Uwe Schröder für Norddeutschland und (06152) 58452, Rainer Monzheimer für Süddeutschland und vieles mehr! Interessenten wenden sich an: Uwe Schröder, Wannerstr. 57, 4650 Gelsenkirchen.

CPC-Compi-Club-User-Gruppe Nord. Achtung, der Compi-Club, Jörg Heise hat nun eine Usergruppe in Norddeutschland. Alle CPC-User im Raume Norddeutschland, die an einer Mitgliedschaft interessiert sind, bitte ich um Kontaktaufnahme. Gerne bin ich bereit Euch ein Informationsblatt zukommen zu lassen. Immer daran denken, ein Club bringt Vorteile, die den Mitgliedern zugute kommen und dabei denke ich nicht nur an enorm günstige Software oder Hardware. Also bis bald. **Kontaktadresse: Uwe Ziegenhagen, Nolde Straße 12, 2350 Neumünster.**

** **NEU! MSX-USERCLUB NEU!** Werden auch Sie Mitglied?!? Keine Beiträge!! Info Bei LCC-Lunar Computer Club Postfach 1542, 4400 Münster. Tel.: Ab 18 Uhr, 0251/616480.

Schneider CPC 464 – User Maluche Lothar / cpu Peter-Dörfler – Straße 70, 8998 Lindenberg, Allgäu, Telefon 08381/7468.

CPU-Bazar Anzeige

Stark reduzierte Computerbücher aus Remissionen

Spaß mit Basic, 6. Aufl.
Das erfolgreiche Basic-Lehrbuch
statt DM 29.80 nur DM 10,–

Spaß mit BASIC für Anwender
Nützliche, nicht maschinenabhängige Programme
statt DM 26,– nur DM 9,–

Spaß mit Basic für Profis
Programme aus der Praxis, nicht maschinenabhängig
statt DM 26,– nur DM 9,–

Computer für jedermann ZX 81 + Spectrum
statt DM 32,– nur DM 11,–

Basic mit Pfiff
Eine Fülle von Programmbeispielen und -tricks
statt DM 32,– nur DM 11,–

PET/CBM Spaß und Spiele
Über 30 herausragende Spiele
statt DM 34,– nur DM 10,–

In 60 Minuten zu Computerwissen
statt DM 21,80 nur DM 8,–

Kinder lernen spielend programmieren
statt DM 24,80 nur DM 9,–

Basic leicht gelernt
statt DM 26,– nur DM 10,–

Der perfekte Führer zu Ihrem IBM Computer
statt DM 34,– nur DM 14,–

BASIC SOFTWARE BIBLIOTHEK
Nicht maschinenbezogene, nützliche Programme
Bd. 1-4, statt DM 18,– nur DM 8,– je Bd.
zusammen DM 30,–

BASIC lernen am praktischen Beispiel in MBASIC 1
Programme für den Hausgebrauch
statt DM 34,– nur DM 14,–

20 lehrreiche Spiele für den VC 20
Superspiele statt DM 21,80 nur DM 10,–

Spielspaß für den VC 20
30 außergewöhnliche Spiele
statt DM 21,80 nur DM 10,–

Spielspaß für den CBM
über 30 tolle Spiele
statt DM 34,– nur DM 10,–

Computerwissen für Einsteiger
statt DM 29,80 nur DM 10,–

Nur solange Vorrat reicht. Kaufen Sie jetzt günstige Weihnachtsgeschenke. Lieferung per Verrechnungsscheck oder Nachnahme.

Bestellungen richten Sie an:
HAPA Buchdienst
Brucker Straße 46 · 8031 Gilching

Autoren-Wettbewerb in „Tips u. Tricks von CPU:“ Der Leser wird mitentscheiden, wer gewinnt!

Das CPU-Team war schier „aus dem Häuschen“: Mit solch' einer Resonanz durch Sie, den Leser, hatte man nicht erwartet. Jetzt geht's in die zweite Runde. Und: Der Leser ist aufgerufen mitzuwirken!

Tolle Gewinne warten auf Ihre Gewinner: Insgesamt **600 Deutsche Mark** *werden jeden Doppelmonat für die besten Autoren-Beiträge „an den Mann gebracht“. Die Wettbewerbs-Kriterien: Der Wettbewerb gilt für die besten Einsendungen in den Rubriken*

1. User (maximal drei Seiten sollten bei uns „ankommen“)

2. Utility (hier sollten es max. zwei Seiten sein)

3. Tips & Tricks (hier maximal eine Seite)

Der Leser – als Juror – wird mitentscheiden!

Das Team der CPU wird dann die besten Einsendungen und Tips wie folgt honorieren:

für den besten „User-Text“ = 300 Mark,

für die beste „Utility“ = 200 Mark und

für die besten Tips & Tricks = 100 Mark

Sie sehen also, Mitmachen lohnt sich! Nur, wer mitwirkt, kann gewinnen. Und: Mehr noch – der „Spitzen-Autor“ wird in der nächstfolgenden Ausgabe der CPU vorgestellt.

Wenn Sie sich an diesem Wettbewerb beteiligen wollen, übersenden Sie uns bitte folgende Unterlagen:

1. **Ausführliche Beschreibung des eingesandten Beitrages**
2. **Datenträger (Kassette oder Diskette)**
3. **Komplettes Listing (wenn zur Beschreibung erforderlich)**
4. **Für die Autorenvorstellung: Infos zur eigenen Person (kurzer Lebenslauf, Hobbies etc.) + Foto**
5. **Frankierten Freiumschlag (mind. 1,40 DM)**

All dies senden Sie bitte an den TRONIC-Verlag, Postfach, 3444 Wehretal 1.

Also, mitmachen, mitwirken und mitgewinnen! CPU – das Magazin mit Pfiff!

DOPPELT GEWINNEN MIT "ELEPHANT"!

ELEPHANT MEMORY SYSTEMS™

"NEVER FORGETS."

Jetzt gibt es neben dem 1. Gewinn, der überragenden Qualität der "ELEPHANT"-Floppy-Disk, noch einen triftigen Grund mehr, schnell Ihren "ELEPHANT"-Händler aufzusuchen: ein Puzzle-Spiel, das Ihnen einen schönen Extra-Gewinn einbringen kann. Was es zu gewinnen gibt, steht auf dem fertigen Puzzle!
Also nichts wie hin zu Ihrem "ELEPHANT"-Spezialisten.

Dennison

ELEPHANT™ NEVER FORGETS.

MARCOM Computerzubehör GmbH
Podbielskistraße 321, 3000 Hannover 51, Tel. (0511) 64 74 20

Frankreich: Soroclass, 8, Rue Montgolfier · 93115, Rosny-Sous-Bois, Tel.: 16 (1) 855-73-70
Großbritannien: Dennison Mfg. Co. Ltd., Colonial Way, Watford WD2 4JY, Tel.: 0923 41244, Telex: 923321
Weiteres Ausland: Dennison International Company, 4006 Erkrath 1, Matthias-Claudius-Straße 9, Telex: 858 6600